Wireless Chord Creator for Guitars with Pick-ups

By

Edward Michael L. Abad
Karen B. Cornejo
Rachelle G. Santos

A Design Report Submitted to the School of Electrical Engineering, Electronics and Communications Engineering, and Computer Engineering in Partial Fulfilment of the Requirements for the Degree

Bachelor of Science in Computer Engineering

Mapúa Institute of Technology
November 2008

Approval Sheet

Mapúa Institute of Technology
School of EE-ECE-CoE

This is to certify that we have supervised the preparation of and read the design report prepared by **Edward Michael L. Abad, Karen B. Cornejo** and **Rachelle G. Santos** entitled **Wireless Chord Creator for Guitars with Pick-Ups** and that the said report has been submitted for final examination by the Oral Examination Committee.

Prof. Susana T. Alabastro
Reader

Engr. Cyrel C. Ontimare
Design Adviser

As members of the Oral Examination Committee, we certify that we have examined this design report, presented before the committee on **November 24, 2008,** and hereby recommended that it be accepted as fulfilment of the design requirement for the degree in **Bachelor of Science in Computer Engineering.**

Mlatina
Engr. Mary Ann Latina
Panel Member

Engr. Vic Dennis Chua
Panel Member

Engr. Mary Jane Quinit
Chairman

This design report is hereby approved and accepted by the School of Electrical Engineering, Electronics and Communications Engineering, and Computer Engineering as fulfilment of the design requirement for the degree in **Bachelor of Science in Computer Engineering.**

Felicito S. Caluyo
Dr. Felicito S. Caluyo
Dean, School of EE-ECE-CoE

ACKNOWLEDGEMENT

We, the designers, would like to thank our parents for the support that they have given us in building the design that we have made; all our friends who willingly participated in testing our design effectively; Engr. Cyrel C. Ontimare who has been patient in giving us lectures and recommendations in the making of the design; Prof. Benigno Agapito Jr. who help us in fixing the format for our documentation; Prof. Susana T. Alabastro who patiently guided us in constructing our paragraphs and checking the grammar of our documents; and most especially God, who gave us the strength and patience in constructing our design and our documentation.

Edward Michael L. Abad
Karen B. Cornejo
Rachelle G. Santos

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

Wireless Chord Creator using Pick-ups is a portable device that is used to convert notes into chords and display them afterwards. This design is used by guitarists to easily recognize the chords that they have performed. In order for the gadget to work a phase lock loop (PLL) and a PIC microcontroller to display the notes and its equivalent chord are used. The system uses RF through Wireless FM Transmitter and Receiver since it is designed to be wireless for the convenience of the user.

Keywords: PIC, notes, chord, cord, octave, guitar, pick-up, tune, strum, pluck, resonance, fret

Chapter 1

DESIGN BACKGROUND AND INTRODUCTION

a. The Design setting or context or frame of reference

Chords are the combination of notes that make up a distinguishing sound or tune. In most cases, guitarists whether beginners or professionals make up chords that are hard to explain. There are circumstances when they may have forgotten or cannot explain what they are called because they have just made up these chords accidentally. In order for others to understand what chord is, the guitarist who only made up that chord tries to identify it note by note, but by doing so, consumes much time.

Guitar chords are usually published in magazines, songbooks, and even the internet. Other formation of chords is not included in these resources; thus giving the guitarist the freedom to express each chord in his/her own way or technique.

When playing the guitar with friends or band-mates, the guitarist might be asked about the chord play. Since the guitarist cannot explain his/her own shape of chords, he/she must go into details for others to understand the chord. There is a need for a device or gadget that will help solve this problem.

b. Statement of the Problem

Since chords are essential to musicians, it is important to understand easily how chords are made. Guitarists have a tendency to make up chords that sounds like the chord they want to produce but with a different "feel" into it, thus making it hard for them to tell what chord that is. With this gadget, the combination of notes pressed on the guitar fretboard that is plucked or even strummed will be given a clear definition of a chord.

c. The Objective of the Design

The group aims to create a device that will help guitar players remember the chord they are playing as well as help boost their creativity in making chords. They also considered the following to be able to implement the specified gadget:

1. To be able to interface an RF to the Wireless Chord Creator for it to work at a distance.
2. To be able to convert each combination of notes that is plucked or strummed on the guitar and output it as a chord on a display.
3. To be able to make the design easy to use.

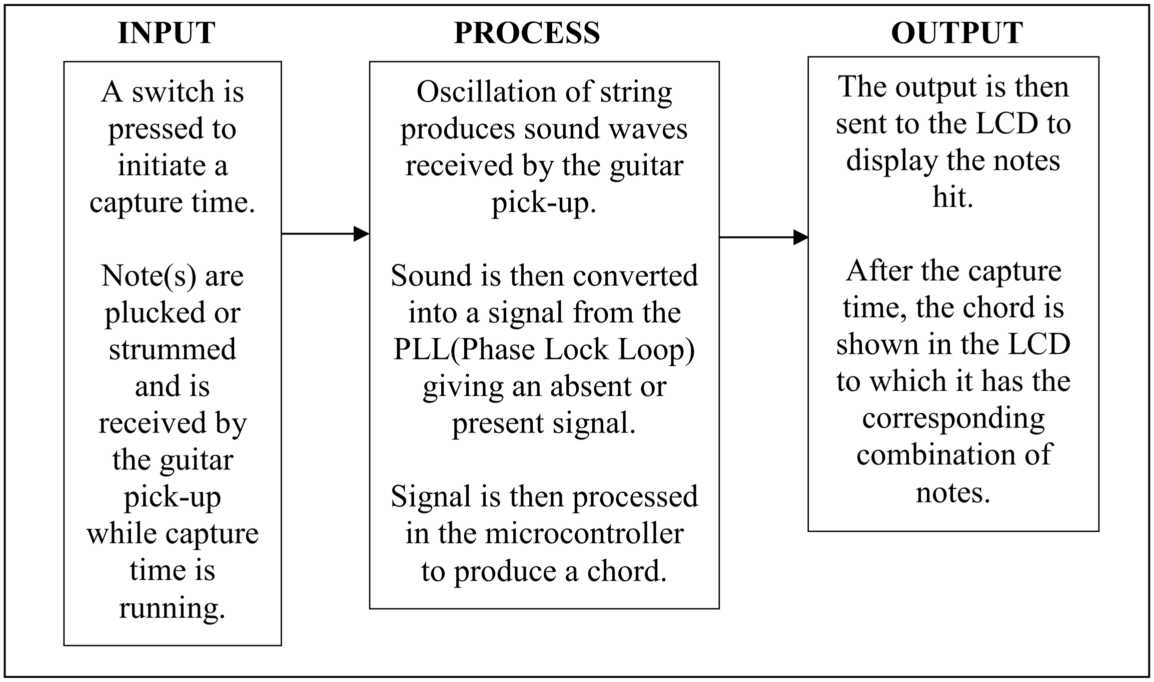
d. The Significance of the Design

The importance of this study is to provide guitarists or guitar players a guide to the chords they are playing, giving them the idea that the notes they chose is referenced in a real chord. This will benefit guitar players when they are teaching others or even practicing by themselves or with their band due to the ease of knowing the chords they play. It will also lessen the amount of time spent in going into details when explaining a simple chord. This can improve the creativity of a person by enabling him/her to practice each combination of notes that leads into playing a chord.

It will help band-mates as well as friends by giving them the idea what chords are used in the song being played. Instead of asking what chords are being played, they will just follow the output sent by the guitar. They may even use a different chord shape that they like. This will be a necessity for guitar players who forget chords. They will just follow the lead of the guitarist and refer to the output the gadget will produce. They can try to make-up their own chord.

e. The Conceptual Framework

In order to construct the design, the group talked about certain ideas related to this study. After the brainstorming session, each member agreed on one thought which resulted in one concept. Figure 1.1 illustrates the flow on which the device will work. It covers the three major parts such as the Input, Process and Output. It is a brief overview of the features of the device.



A switch is pressed to initiate a capture time.

Note(s) are plucked or strummed and is received by the guitar pick-up while capture time is running.

Oscillation of string produces sound waves received by the guitar pick-up.

Sound is then converted into a signal from the PLL(Phase Lock Loop) giving an absent or present signal.

Signal is then processed in the microcontroller to produce a chord.

The output is then sent to the LCD to display the notes hit.

After the capture time, the chord is shown in the LCD to which it has the corresponding combination of notes.

Figure 1.1: Conceptual Framework

Concept Model

Using the concept model found on Figure 1.1, the design shows that when a note or a combination of notes is strummed or plucked, it will be received by the guitar pick-up. Before plucking or strumming the strings, the switch should be set first to have ample time to capture the notes. The signal then travels to the device PLL (Phase Locked-Loop) which process the audible signal in an absent or present state like a digital signal either as 0 or 1. The process of deciphering the chord by means of the combination of signals (notes) will take place in the microcontroller and will output a signal that is sent to the LCD display, which is not attached to the gadget. It can also be placed at a distance estimated to be within a typical room size for the user to see it from afar. The LCD with the main board can be attached to the guitar itself or placed at a certain distance not greater than 10m for other users to see the chords being played. The LCD then shows the output of the notes with the chords the user has just played.

f. The Scope and Delimitation

The device covers and delimits to the following:

Scope

1. The device will use RF technology.
2. It can be easily attached or un-attached to a guitar.
3. The output will be displayed on a LCD screen.
4. The gadget will have two separate parts; one part is for the display (LCD) as well as the main board for the processing of signals, the other part is for the output of the guitar that will send the signal to the main circuit.
5. The amplifier connected to the transmitter has an added slot wherein you can connect a microphone cable on one end and the other end to an amplifier system if the user wishes to.
6. A push-button switch is used to trigger the start of processing of the system design.
7. It will use the PLL (Phase Locked-Loop) principle.

Delimitation

1. It can only be used for a guitar with a guitar pick-up.
2. The design cannot show the positioning of notes to be pressed as a chord is played.

3. It cannot be submerged in water or any form of liquid.
4. It is more accurate if the guitar and strings are in good working condition.
5. The guitar should be in standard tuning to have better results.
6. The circuit only covers 3 octaves from 110 Hz (A Note) to 830.6094 Hz (G# Note).
7. Chords created are only the common chords used; Major, Minor, Suspended, Seventh, Major Seventh, Minor Seventh, Diminished, and Augmented.
8. It is battery operated.
9. Its wireless capability is up to a maximum of 10 meters.
10. The main board is connected to the LCD screen by means of a cable of 2 meters.
11. The transmitter must be set to 107.1 MHz since it is the pre-set default frequency; thus, it can be varied if the receiver is calibrated again in synchronization with the transmitter.
12. The main board only uses 1 rocker switch for the On/Off of the power supply.
13. It cannot output multiple chords at a time.
14. It only has a capture time of 8 seconds.

g. Definition of Terms

1. **Amplifier** - sound-increasing apparatus: a device that makes sounds louder, especially one increasing the sound level of musical instruments *(Encarta® World English Dictionary).*
2. **Chords** - notes struck together: two or more musical notes played or sung simultaneously *(Encarta® World English Dictionary).*
3. **Circuit** - route for electricity: a route around which an electrical current can flow, beginning and ending at the same point *(Encarta® World English Dictionary).*
4. **Cord** - electrical cable: flexible insulated electric cable *(Encarta® World English Dictionary).*
5. **Frequency** - rate of recurrence: the number of times that something such as an oscillation, a waveform, or a cycle is repeated within a specific length of time, usually one second *(Encarta® World English Dictionary).*
6. **Fret (Fretboard)** - any of the ridges of wood, metal, or string, set across the fingerboard of a guitar, lute, or similar instrument, which help the fingers to stop the strings at the correct points *(Chicago Manual Style).*
7. **Gadget** - ingenious device: a small device that performs or aids a simple task *(Encarta® World English Dictionary).*
8. **Guitar** - stringed musical instrument: a musical instrument with a long neck, a flat body shaped like a figure eight, and usually six strings that are plucked or strummed *(Encarta® World English Dictionary).*

9. **Guitar Pick-up** - A guitar pickup, also called transducer that converts the vibrations of guitar strings or the guitar body to an electrical signal *(Musician News).*

10. **LCD** – Liquid Crystal Display, an electronic display (as of the time in a digital watch) that consists of segments of a liquid crystal whose reflectivity varies according to the voltage applied to them *(Merriam-Webster).*

11. **Microcontroller** - a microprocessor that controls some or all of the functions of an electronic device (as a home appliance) or system *(Merriam-Webster).*

12. **Music** - sounds that produce effect: sounds usually produced by instruments or voices that are arranged or played in order to create an effect *(Encarta® World English Dictionary).*

13. **Note** - musical or vocal sound: a sound of a distinct pitch, quality, or duration produced by a musical instrument or by the voice *(Encarta® World English Dictionary).*

14. **Octave** - note at each end of octave: the note at each end of an octave, especially the higher one, considered in relation to the note at the other end *(Encarta® World English Dictionary).*

15. **Phase** - part of repeating cycle: a part of a repeated uniform pattern of occurrence of a phenomenon or process, relative to a fixed starting point or time *(Encarta® World English Dictionary).*

16. **Pluck** - pull and release strings: to play a stringed musical instrument by quickly pulling and releasing strings with a finger or plectrum *(Encarta® World English Dictionary).*

17. **Resonance** - large oscillation at natural frequency: increased amplitude of oscillation of a mechanical system when it is subjected to vibration from another source at or near its own natural frequency *(Encarta® World English Dictionary).*

18. **Signal** - transmitted information: information transmitted by means of a modulated current or an electromagnetic wave and received by telephone, telegraph, radio, television, or radar *(Encarta® World English Dictionary).*

19. **Sound** - reproduced music or speech: the music, speech, or other sounds heard through an electronic device such as a television, radio, or loudspeaker, especially with regard to volume or quality *(Encarta® World English Dictionary).*

20. **String** - cord stretched across musical instrument: a cord made of nylon, wire, or gut that is stretched across a musical instrument and plucked, bowed, or otherwise vibrated to produce sound *(Encarta® World English Dictionary).*

21. **Strum** - play instrument by brushing strings: to play a guitar or other stringed instrument by brushing the strings with the fingers or a pick *(Encarta® World English Dictionary).*

22. **Technology** - application of tools and methods: the study, development, and application of devices, machines, and techniques for manufacturing and productive processes *(Encarta® World English Dictionary)*

23. **Tune** - adjust instrument for pitch: to adjust a musical instrument so that it plays at the correct pitch *(Encarta® World English Dictionary).*

24. **Wireless** - using radio signals: using radio signals rather than wires *(Encarta® World English Dictionary).*

Chapter 2

REVIEW OF RELATED LITERATURE AND RELATED STUDIES

The concept of making this type of system design was brought up when the designers came across certain studies while conducting the research. They worked together on the ideas of existing studies and utilize them in their design.

An article by Peter Gitundu in Music, Recreation and Leisure, Art and Culture magazine entitled, "Why You Should Be Using An Electronic Tuner Today" stated that when a guitar is tuned up, the stress tension is changed on one string at a time. This tuning applies to all strings whether one is using an electronic guitar or not. Nylon strings may just take a little more effort to settle into tune. This is a significant information because it means that a certain string corresponds to a certain tension. A tension will lead to how much the string or group of strings oscillate that will produce a unique frequency that can be based on a reference.

It is necessary to know the key in which the guitar needs to be tuned. Normally for a 6-string guitar basic keys of EADGBE are used. Hence if you need to tune the guitar in standard form, it is not necessary to change tuning keys because it sounds in EADGBE. These notes or keys can be adjusted by tightening or loosening each string.

An electronic tuner makes tuning very much easier. This principle can be applied to the design since chords are a combination of notes, and in order for one to determine a note a tuner is needed. If a string is strummed or plucked, guitar’s knobs can be turned on until the guitar strings match with the corresponding pitches of the instrument.

The article helped the designers to conceptualize that tuning instruments have a reference which can be called a “standard”. This means that one note must be in the same frequency caused by the tension of the string of the same note but with a different instrument. It also emphasizes that electronic tuners are much easier to use thus giving the designers a push to pursue the creation of an electronic device.

Wireless technology is the approach used in the design. Making the device use this type of technology broadens the scope of the design. In an article published in the Modern Guitars Magazine (January 9, 2007), X2 Digital Wireless, Inc. announced that the XDS95 is the first and only digital multi-channel digital UHF wireless system designed specifically for performing or recording musicians and is now shipping through authorized dealers. This gadget uses digital RF modulation, hence the idea of using RF technology is one of the options. Since the XDS95 is advertised as the first and only wireless system for musical applications, no other device in the music industry have the same approach; thus giving the proposed design a slight edge. Figure 2.1 shows the XDS95 which comprises of two parts: the receiver and the transmitter. Since it uses RF technology, it is a given that these two parts are needed to make gadget work.

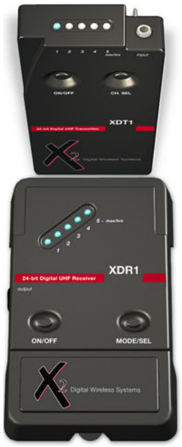
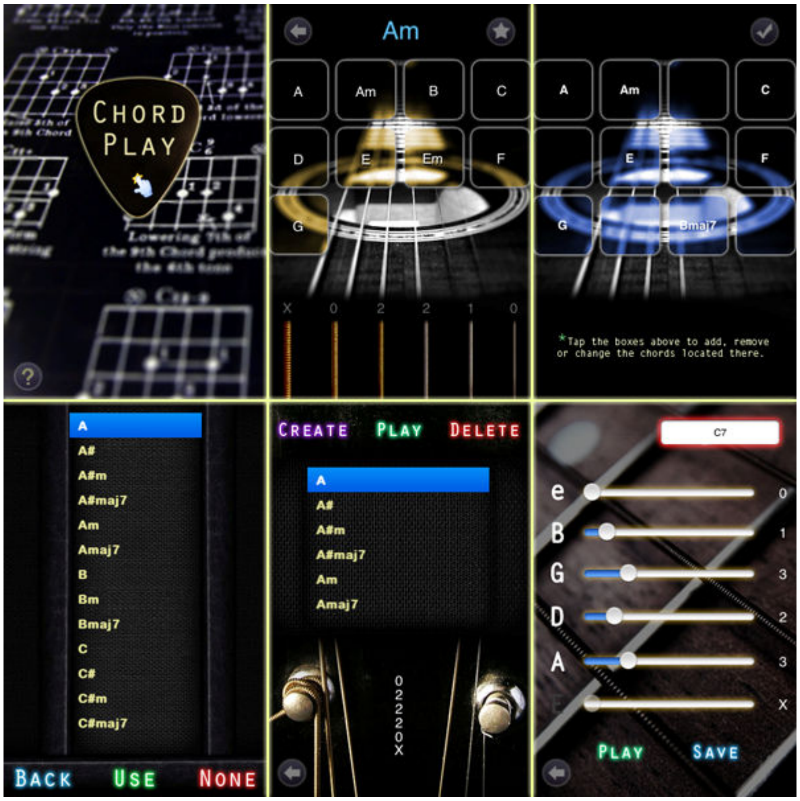


Figure 2.1: XDS95

The chord plays a vital role in making the device. It is necessary that the said tool can output an expected chord. The article by Jody Mitoma of Touch Podium entitled, "'Chord Play' Lets You Play up to 12 Guitar Chords at Any Given Time" (August 20, 2008) is about computer software, which lets you play guitar chords on your iPhone and iPod Touch. It is done by simply tapping on the chords you want to play and then use the six strings to play the notes of the chord as you would on a guitar. The chord creator lets you define and save even more chords. This kind of software lets you define the chord you want to produce. This information helped the designers figure out the main features of the device. The chords the user defines in the guitar will be displayed on a screen. This will give the user the freedom to create new chord positions he/she want to use. Figure 2.2 shows the software display in use. It features defining a chord by means of showing the notes on the top center of the screenshot which gives the designers an idea on how to present the output of the device. The group figured out that the output should display the notes of the chord as well as the main part which is a chord.



*Tap the boxes above to add, remove or change the chords located there.

Figure 2.2: Screenshot of ‘Chord Play’

Mounting the device is one of the group’s discussions wherein different ideas came into play. An article by Conner Flynn on Peak (January 11, 2008), entitled “eNote Clip-On Digital Chromatic Tuner” discussed the use of a clipping device. It’s as simple as to just attach the little device to the guitar or other instrument and the display will show up green once the note is correct. Clipping the mechanism to the instrument made it easy to use.

The idea of using an LCD screen for the device was inspired by the article by Lou Reade of Innovative Engineering Device (October 21, 2007). The article is about a product launched at the Frankfurt Music Show that helped guitarists to solve one of their greatest headaches, tuning.

The device, the S440 tuner, was developed by Somerset-based ATD. It displays the output on a LED screen. Since Maxon Motor UK developed the product based on LED, an LCD screen was used in the proposed design for a clear and elegant display. Figure 2.3 shows the S440 Tuner that is attached to the guitar. Its display can be seen on a LED screen, while its output will be incorporated on a LCD screen. However, a larger screen will be used so that other essential information could be squeezed in.

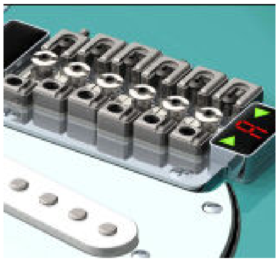


Figure 2.3: S440 Tuner

Microcontrollers are very popular in implementing electronic devices. This concept was applied to the design since it is widely used in most parts of the country. An article by Ariz Chandler of CPU Technologies (March 3, 2003), entitled “New Microchip PIC16 New Low-Power Microcontrollers with nanoWatt Technology” described the great features of the new microchip PIC16 that will boost the performance of such a device. It offers the flexibility of re-programmable Flash memory coupled with new power management features, and are designed to reduce the overall power consumption in embedded systems. This feature of the PIC gives a solid foundation of components to make the design possible.

The IC is affordable and easily available here in the Philippines. This information is important as it makes the design possible.

Chapter 3

DESIGN METHODOLOGY AND PROCEDURES

Design Methodology

Wireless Chord Creator for Guitars with Pick-Ups is a system design that provides guitarists or guitar players ease of determining the chords they are playing. This design is the first of its kind since other guitar related devices which are guitar tuners name only one note to be played.

Applied research was used to solve practical problems that relate to this kind of study. This form of research is necessary to improve on this field of technology. Inquiries from other people as well as using books as references are key methods of understanding the problem itself. With this, one's knowledge about this area of study can broaden and open to new ideas to enhance the gadget. Rigorous reading from different sources such as books, magazines and other materials contribute to a better understanding of the subject at hand. Having all the information formulates in the development of the actual device. Data attained by the group is utilized to create such a device.

Design Procedure for Actual Design

The designers have taken a step by step procedure in making the whole hardware design. These steps are as follows;

1. The first step was to gather information from related studies to have adequate background of the area of the study. It is vital to know other information related to the study because it will help in the understanding of

the concept of the design. Conceptualization was also done in this part by brainstorming among the members of the group. One of the key data gathered was the Musical Note Frequency Table shown in Table 3.1 below.

Key Notes	OCTAVE										
	1	2	3	4	5	6	7	8	9	10	11
A		27.5	55	110	220	440	880	1760	3520	7040	14080
Bb		29.14	58.27	116.54	233.08	466.16	932.33	1864.66	3729.31	7458.62	14917.24
B		30.87	61.74	123.47	246.94	493.88	987.77	1975.53	3951.07	7902.13	15804.27
C		32.7	65.41	130.81	261.63	523.25	1046.5	2093.01	4186.01	8372.02	16744.04
C#		34.65	69.3	138.59	277.18	554.37	1108.73	2217.46	4434.92	8869.84	17739.69
D		36.71	73.42	146.83	293.66	587.33	1174.66	2349.32	4698.64	9397.27	18794.55
D#		38.89	77.78	155.56	311.13	622.25	1244.51	2489.02	4978.03	9956.06	19912.13
E	20.6	41.2	82.41	164.81	329.63	659.26	1318.51	2637.02	5274.04	10548.08	
F	21.83	43.65	87.31	174.61	349.23	698.46	1396.91	2793.83	5587.65	11175.3	
F#	23.12	46.25	92.5	185	369.99	739.99	1479.98	2959.96	5919.91	11839.82	
G	24.5	49	98	196	392	783.99	1567.98	3135.96	6271.93	12543.85	
G#	25.96	51.91	103.83	207.65	415.3	830.61	1661.22	3322.44	6644.88	13289.75	

Table 3.1 Musical Note Frequency Table

A particular key or note corresponds to a specific value. Each value in the table is in Hertz (Hz). This is essential because it will distinguish one note from another wherein a chord is a combination of specific notes.

2. Second, the group researched on the ideal or suggested components or parts they can use in doing the design. The group also considered the availability and cost of the components they will use.
3. After canvassing the components to be used, the third step is to design the flowchart was designed to have an overview of how the device will work.
4. Designing the schematic and circuit diagram was conducted. This was based on the information about the availability of the needed components.

5. PCB designing was made. The components were then integrated to the PCB following their connections from the schematic and circuit diagram.
6. Testing was then conducted after making sure each connection was properly placed. The testing procedure verified if the expected results would occur.
7. If there were still problems, troubleshooting of the design or making other adjustments until the projected results would appear.
8. After this process was made construction took place.
9. Once everything has been completed, a final test with the gadget was made just to make sure everything goes out according to plan.

Hardware Design

1. Block Diagram

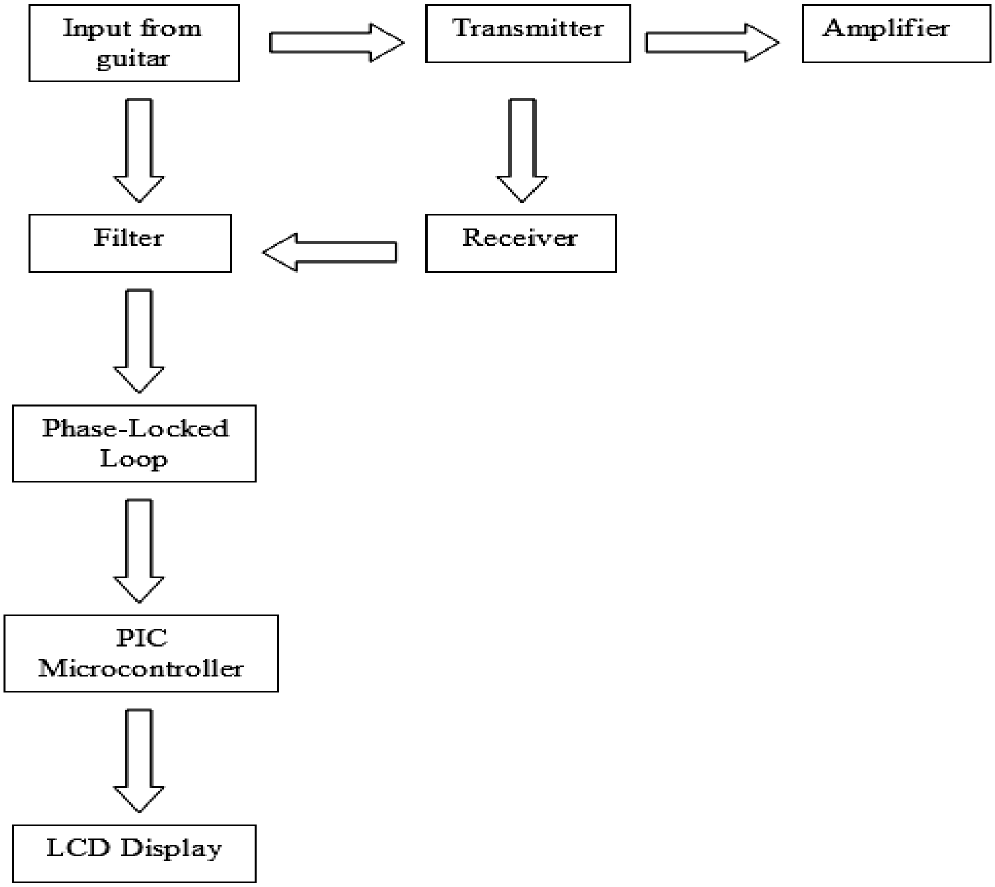


Figure 3.1: System Block Diagram

Figure 3.1 shows an illustration of the System Block Diagram for the system design.

2. Schematic Diagram

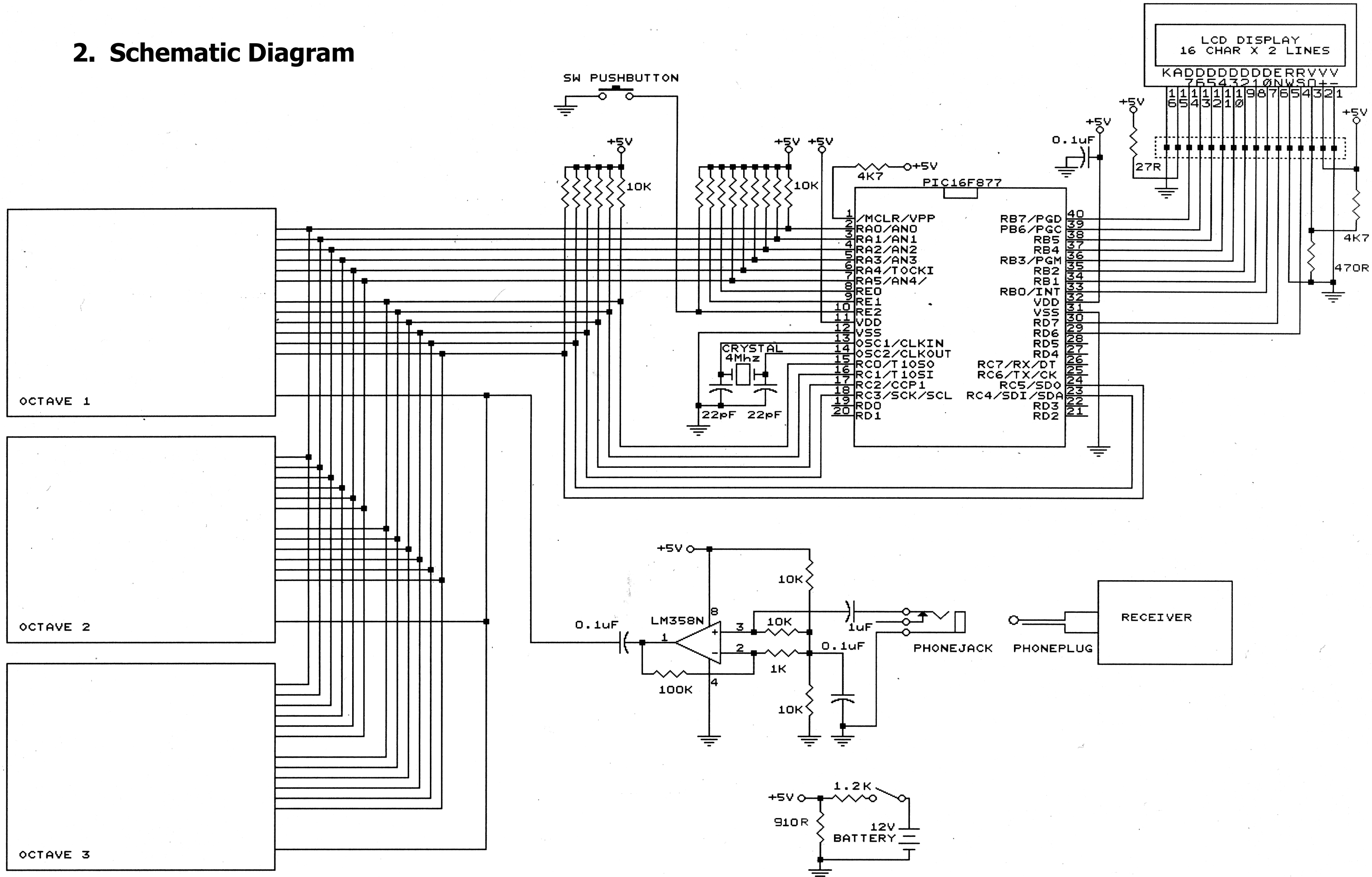


Figure 3.2: Schematic Diagram for the main circuit

Octave 1

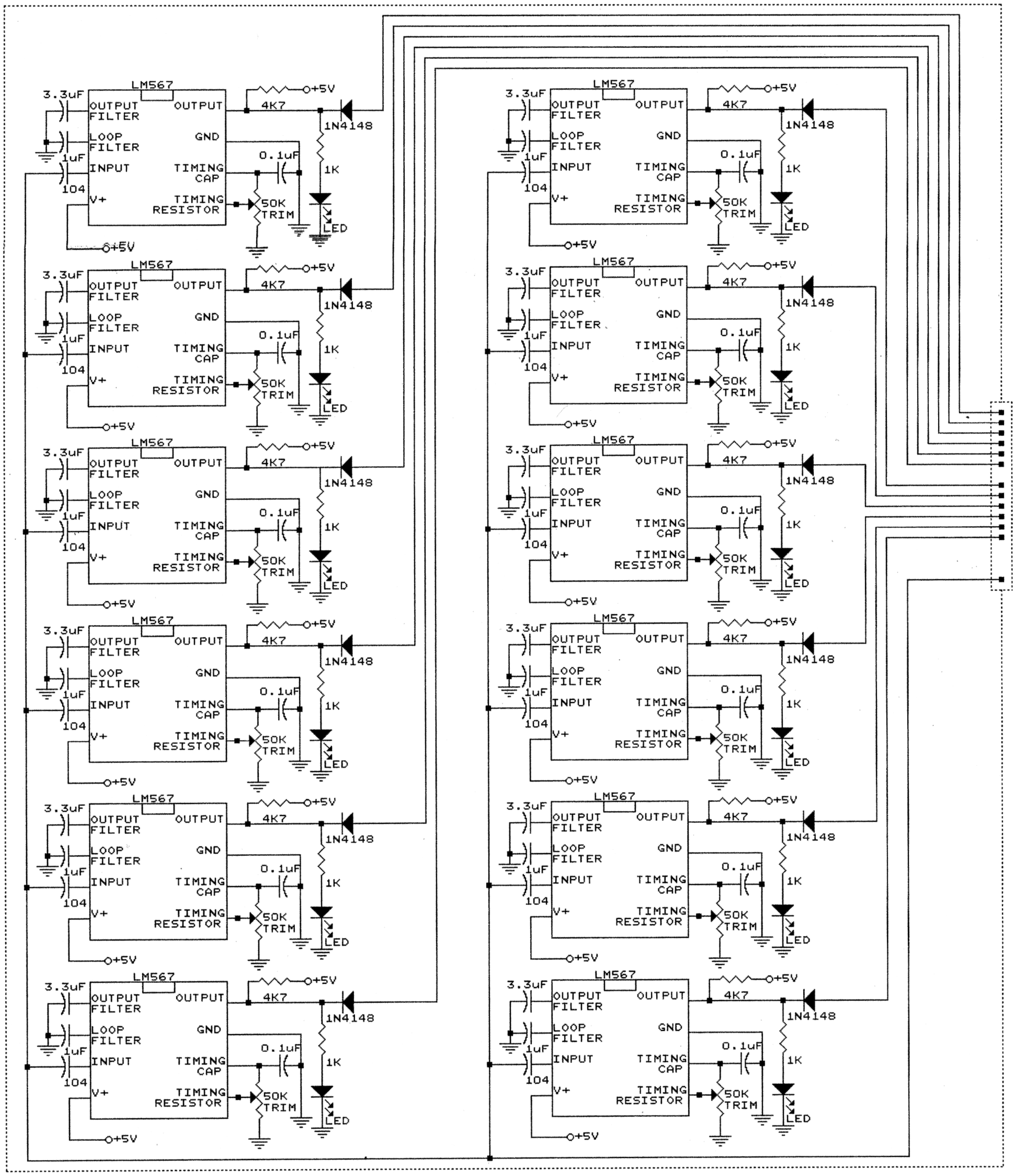


Figure 3.2.1: Schematic Diagram for Octave 1

Octave 2

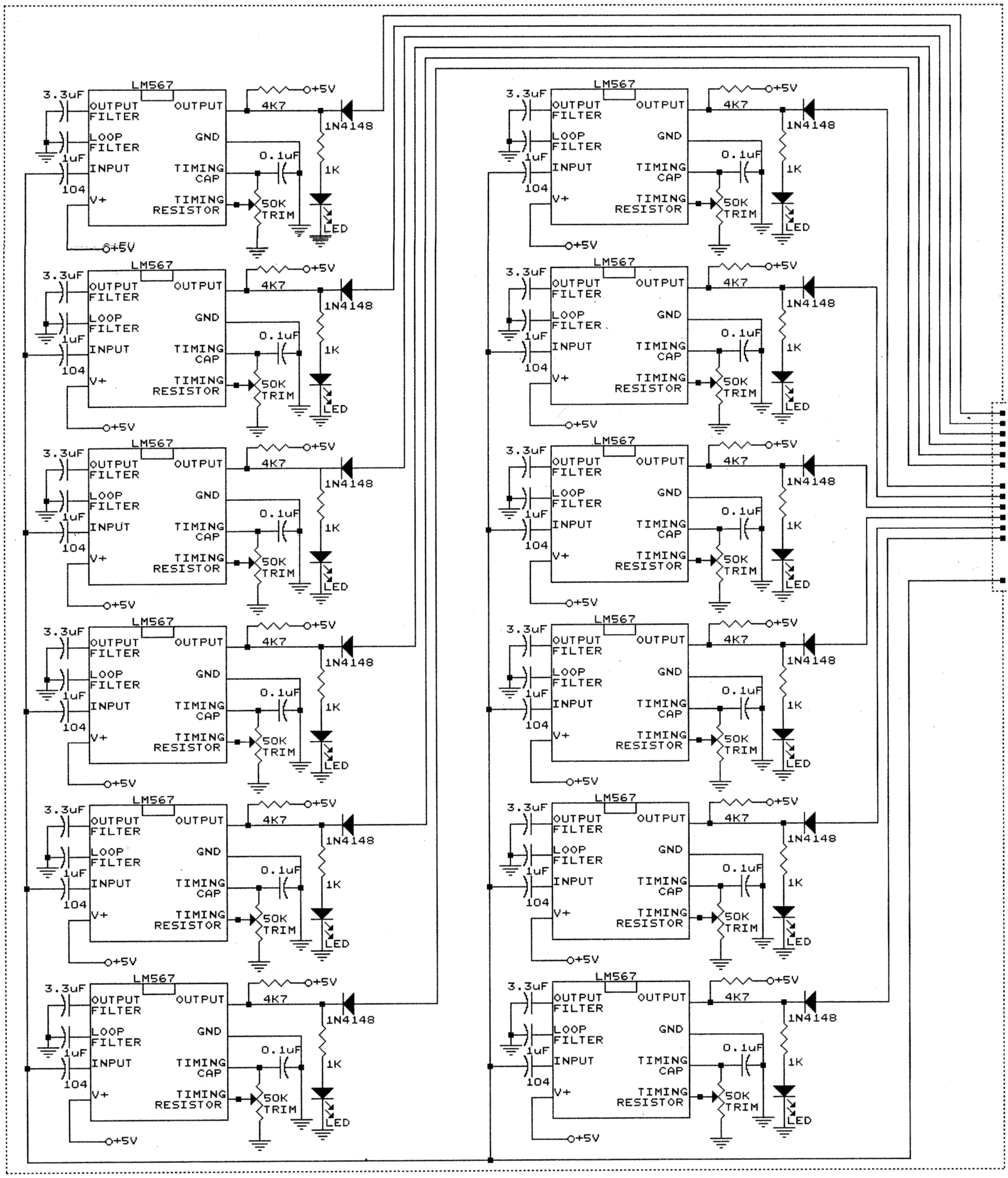


Figure 3.2.2: Schematic Diagram for Octave 2

Octave 3

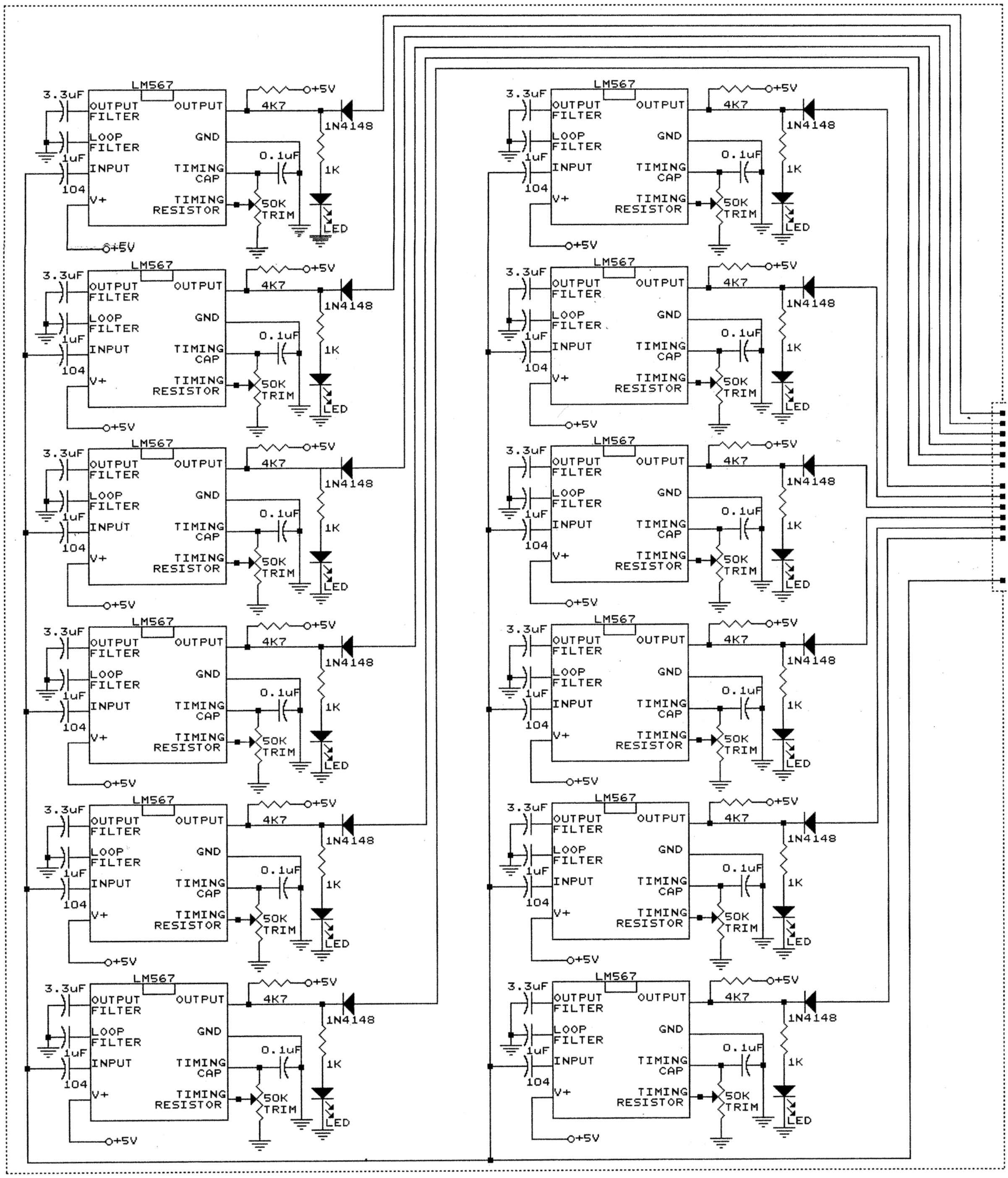


Figure 3.2.3: Schematic Diagram for Octave 3

3. List of Materials

Description	Quantity	Description	Quantity
LCD module 16 character x 2 Line	1	LED	37
Mini Push Button	1	LM567 IC	36
8 pin connector	2	LM358 IC	3
Heatsink	1	Alexan Case Black	1
1200uF/16V electrolytic capacitor	1	Alexan Case White	2
100uF/25V electrolytic capacitor	1	Battery Holder	1
105 multilayer ceramic capacitor	1	Battery AA	8
22pf ceramic capacitor	2	Phone Jack	1
W10G Bridge Diode	1	Rocker Switch	1
1/4Watt resistor	94	IN4148 Diode	36
2 pin terminal block	2	Trimmer resistor 100K	36
4Mhz Crystal	1	8 pin IC Socket	39
10K array resistor	2	FM Receiver	1
40 pins IC Socket	1	Phone Jack Y-adaptor	1
PIC16F877 microcontroller IC	1	Phone Jack converter	1
104 Multilayer ceramic capacitor	48	Microphone Amplifier	1
10uF/16V electrolytic Capacitor	36	Microphone Cable	1
47uF/16V electrolytic Capacitor	36	9V Battery	1
Wireless FM Transmitter	1	bag	1

Table 3.2: List of Materials

Hardware Components

Research was conducted on for the most effective electronic parts for this system design. Some important components of the design are listed below.

PCB Layout

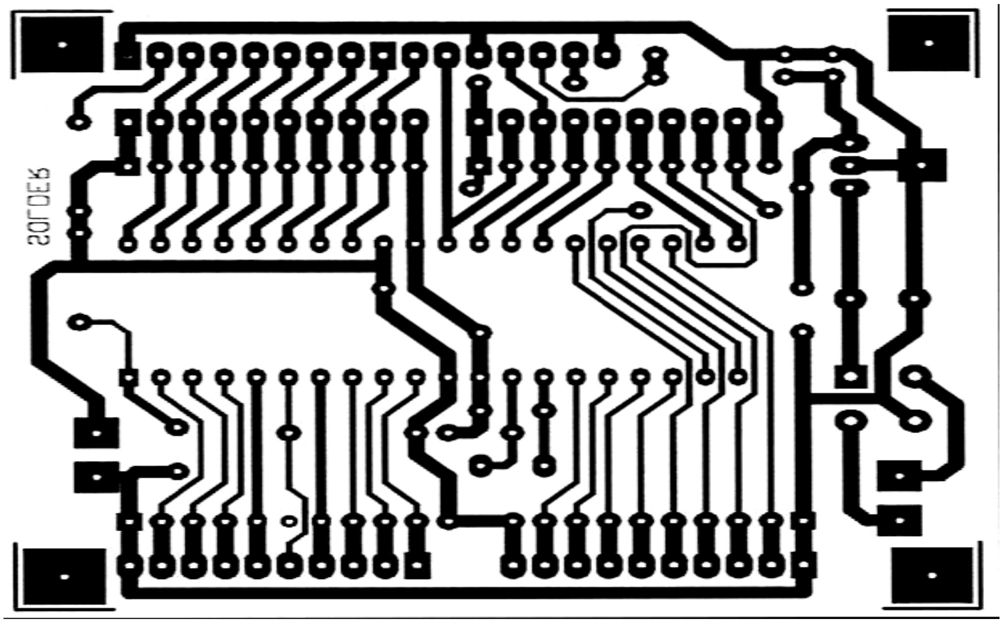


Figure 3.3: PCB Layout for Microcontroller

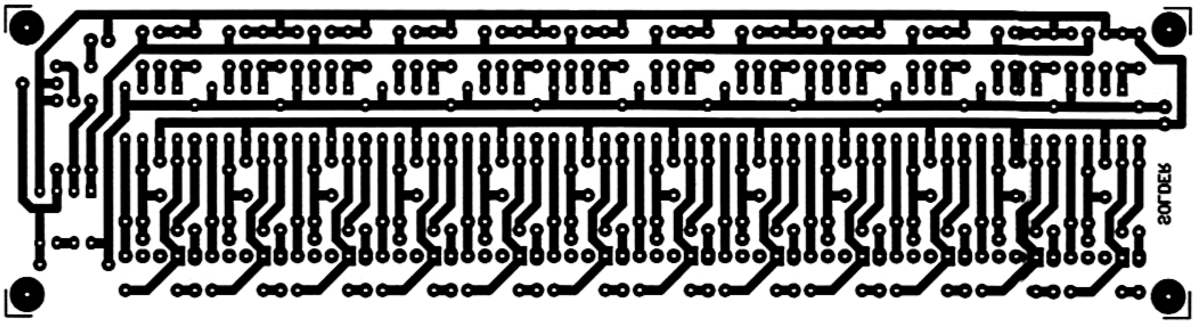


Figure 3.4: PCB Layout for PLL Circuit

Micropower Phase-Locked Loop

Since the design system relies on frequency as an input, the Micropower PLL is the best device for the system design. Micropower Phase-Locked Loop is a device that compares the frequencies of two signals and produces an error signal which is proportional to the difference between the input frequencies. This device will be responsible for receiving and converting the frequency inputted through strumming of the guitar strings. It will output either absent or present which will be passed to the microcontroller as 1 or 0. Each note from three different octaves is embedded with one PLL.

LCD Module

The LCD was used for displaying the output of the system design. The data to be displayed will come from the microcontroller. The LCD will only display the type of chord played by the guitar player. If the input signal is invalid the LCD will display a "Try Again" message. The LCD will also display the countdown of 8 sec. time limit for capturing the frequency of the notes played by the user of the guitar.

Radio Frequency using Wireless FM Transmitter and Receiver

A Wireless FM Transmitter and Receiver were used in order to transmit the input signals created by the guitar to the system design. This device was used to create a wireless connection from the guitar to the system design. This was implemented to provide convenience and allowed the guitarist to move at a maximum distance of 10 meters. It is also possible for other users to utilize a guitar at a certain area in the room and be able to supply an output for the other player to show the chord that user has created.

Software Design

The system designers needed a device for the capturing process and producing the chord created from the strumming of the guitar. A push button was implemented to initiate the capturing process. As a solution, a microcontroller was used to control the capturing process and to produce the chord created.

The microcontroller was used to compare and interpret the converted signals passed by the PLL. The main routine of the program is to capture and produce the chord created by the guitar player. The output is then sent to the LCD to display the created chord.

Software Components

The main software component of the design is the PIC16F877A microcontroller. It is the one responsible for operating the whole system. The converted input from the PLL is passed to the microcontroller to produce chords created. For the program language, the PIC Assembly was used to program the microcontroller. PIC Assembly is much similar to Assembly Language especially in some of its instructions. This language is one of the simplest way to program a microcontroller device.

System Flowchart

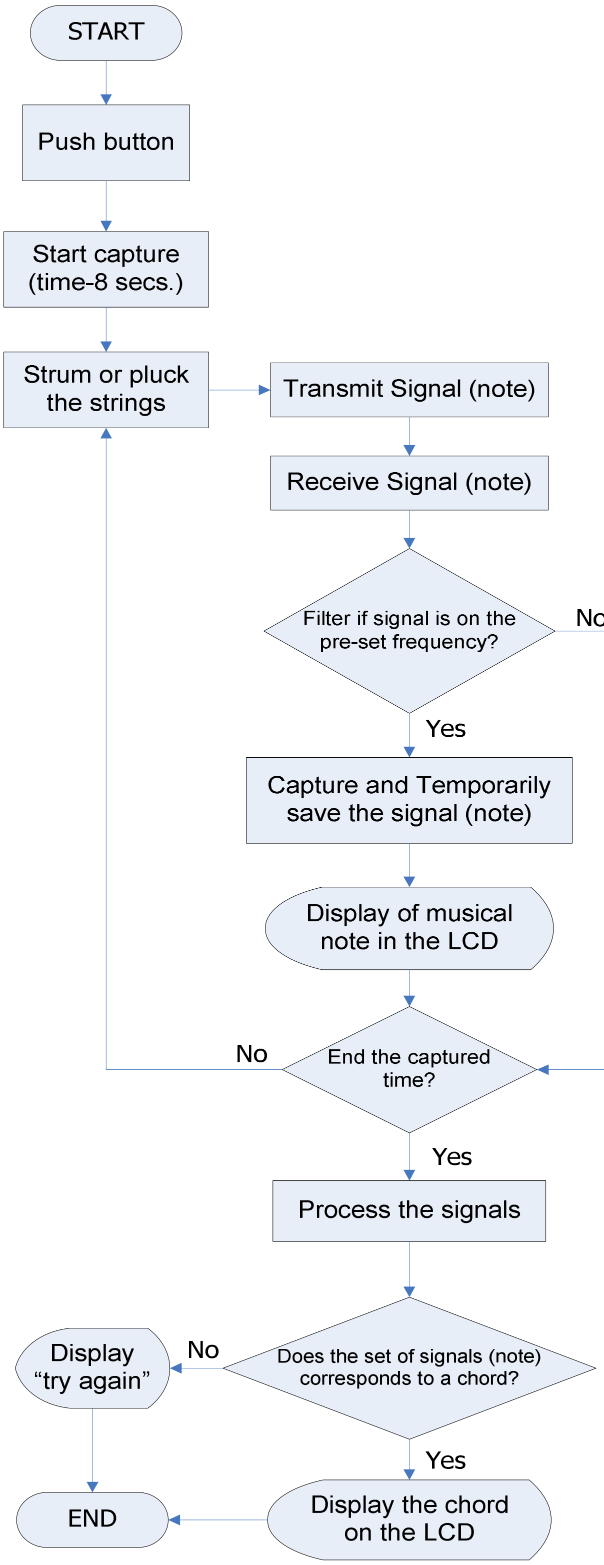


Figure 3.5: System Flowchart

Prototype Development

The following statements summarize the development of the Wireless Chord Creator for Guitars with Pick-Ups.

1. During the first part of the term, the group proposed a project about a device to display the chords created in a guitar. This idea was made possible through research from books, magazines and other materials.
2. Data gathering of related literature and related studies was needed after proposing the project to collect more information to the developers.
3. Research was conducted on how the major components of the device work. The availability and cost of each component were also considered. These components are as follows;
    a) Phase-Locked Loop
    b) Software programs
    c) LCD Module
    d) Radio Frequency Principles
    e) Microcontroller
4. A flowchart was planned and designed to have an overview of how the device will work.
5. The schematic diagram was then created in reference with the flowchart. This is necessary since the flowchart gives a graphical representation of the functions of the device. If the functions are enumerated, the components can then be set depending on which is needed.

6. Using all the information in the schematic diagram, a PCB layout was created. Then the components were then interfaced with each part.
7. The device was tested using a function generator to know if the expected outputs were achieved. The circuit was also calibrated to be set to its purpose.
8. When the setting was completed, the circuit was tested by means of a guitar with pick-ups. This determined if the expected output was correct.
9. When testing was successful, construction of the casing was started.
10. After the completion of the casing, it was again tested if the output or the display was correct with the expected results.
11. Lastly, the maximum distance of the transmitter to the receiver was tested.

Chapter 4

TESTING, PRESENTATION, AND INTERPRETATION OF DATA

Testing the Pre-set Frequency and LED

Testing the circuit if the pre-set frequency is in line with the expected output is crucial. These series of tests shall determine if the signal (note) corresponds to the equivalent value found in Table 3.1 in chapter 3 showing the Musical Note Frequency Table. This is the core reference wherein each signal shall be unique from one another. Since a chord is a combination of notes, these notes have different frequencies which differentiate one from another.

In order to do such test, a function generator is used as a source with the frequency adjusted to a specific value. From the function generator, it is tapped to the PLL circuit with LED present on the output of the circuit. If a signal is set from the function generator and passes through the PLL circuit the LED while light up if the PLL's screw is correctly adjusted. This means that as the group calibrates the value of frequency in the function generator based on the musical note frequency table which assigns one particular frequency to a specific note, each PLL should also be calibrated in line with the signal to have a correct match. There is a total of 12 notes including the sharps and flats in one octave, thus in one octave there is a total of 12 PLL which is assigned to a specific frequency. A total of 3 octaves were used, so 36 PLLs were utilized in the circuit and included in testing.

For every octave, a series of tests were conducted. Shown in Table 4.1 are the results of the initial testing up to the last testing of one octave.

Function Generator (Frequency Set to Corresponding Note)													
NOTE		A	A# or Bb	B	C	C# or Db	D	D# or Eb	E	F	F# or Gb	G	G# or Ab
LED	Trial 1	√	√	X	√	√	√	√	√	√	√	√	√
	Trial 2	√	X	X	√	√	√	X	X	√	√	√	√
	Trial 3	√	√	√	√	√	√	√	√	√	√	√	√

Table 4.1 Function Generator to PLL Calibration

Based on trial 1 of the Table above, the results show that 8 out of 12 notes have a correct match. This means that 8 PLLs are correctly set to allow signals to pass through it at the expected frequency. Trial 2 shows the result of the next series of tests for every note. In that trial, 11 out of 12 notes have a correct match. The third set of trials illustrates that all notes have a correct match with the setting of each PLL. It confirms that the PLL is already set at its target adjustments. These testing procedures for one octave are also done with the two other remaining octaves. The check (**√**) mark represents a correct match from the function generator to the PLL making the LED light up. The (**X**) mark indicates that the LED did not light up, meaning the PLL is not matched to the setting of the functions generator. Its results explain that in the first set of trials, 66.67% (8/12) notes are correct. In the second set, 91.67% (11/12) notes are correct set to the PLL. Lastly, in the third set, all notes are 100% (12/12) correct in reference to the functions generator to the PLL's setting.

Testing the Pre-set Frequency and Guitar with pick-up

Using the guitar as replacement of the function generator is the next step in testing if the correct range of frequency covers the pre-set PLL. By using the guitar, the strings are plucked or strummed to produce a signal that will determine if the octave used in the Musical Note Frequency Table (Table 3.1 Chapter 3) is sufficient. Since the actual application of the design is using a guitar, it is best suited to test the circuit with it.

A guitar with pick-up has a slot which is connected from the guitar pick-up and attached to the body of the guitar. The slot is used for guitar cables to which the cable can be connected to an amplifier system. In this scenario, the cable is plugged into the circuit so that each note in the guitar can be tested if it matches the PLL setting to light up the LED. Each note is plucked one at a time to verify if it lights up the LED corresponding to its own PLL. If the LED that corresponds to the note lights up it means that the PLL was set correctly, but if it doesn't more tweaking is necessary. If the frequency is not covered in the chosen octave, the octave used is adjusted up to which the notes in the guitar are satisfied.

Table 4.2 presents the results of tests conducted that verifies if the guitar notes matches the notes (pre-set Frequencies) of each PLL. A check (**√**) shows that a LED lit up meaning the expected output is correct; an (**X**) mark indicates the LED didn't light up due to incorrect PLL setting.

Guitar													
NOTE		**A**	**A#** or **Bb**	**B**	**C**	**C#** or **Db**	**D**	**D#** or **Eb**	**E**	**F**	**F#** or **Gb**	**G**	**G#** or **Ab**
LED	**Trial 1**	**X**	**X**	**X**	**X**	**X**	**X**	**X**	**X**	**X**	**X**	**X**	**X**
	Trial 2	**X**	**X**	**X**	**X**	**X**	**X**	**X**	**√**	**√**	**√**	**√**	**√**
	Trial 3	**X**	**X**	**X**	**X**	**X**	**X**	**√**	**√**	**√**	**√**	**√**	**√**
	Trial 4	**X**	**X**	**X**	**X**	**√**	**√**	**√**	**√**	**√**	**√**	**√**	**√**
	Trial 5	**X**	**X**	**X**	**√**	**√**	**√**	**√**	**√**	**√**	**√**	**√**	**√**
	Trial 6	**√**	**√**	**√**	**√**	**√**	**√**	**√**	**√**	**√**	**√**	**√**	**√**

Table 4.2 Guitar with pick-up to PLL Calibration

In these series of tests, each note included in the table corresponds to all octaves in the guitar. This means that a certain note in the table can represent 3 octaves of the guitar's notes. In short, each column of notes is the average of all octaves that a particular note touches. Since the guitar's first 12 frets, counting from the end of the guitar's neck away from the body is the same as the 13th fret onwards the testing covered is only the first 12 frets considering it as a reference. To make it clear, the 13th fret is the same as the 1st fret. The table above was used as a summary of tests considering all octaves. In trial 1, all LEDs did not light up because most of the notes considering the octaves did not match. It means that the chosen octaves (basing from the Musical Note Frequency Table) are not the suggested octaves a guitar covers. It shows that 0 out of 12 (0/12) 0% LEDs did not light up. The octaves were adjusted by choosing one octave higher which results in trial 2. Again, it did not match up perfectly showing 5 out of 12 (5/12) 41.67% success rate. In trial 3 one increment of octave is needed to best suit the range. Seeing that trial 3 has 6 out of 12 (6/12) 50% success rate the group verified that the octaves they needed was Octave 4, 5, and 6 from the Music Note Frequency Table. It corresponds to a frequency range of 110Hz to 830.61Hz. After choosing the right range, trial 4 and trial 5 had discrepancies only due to the PLL settings which needed a little tweaking. The table illustrated a 66.67% 8 out of 12 (8/12) and 75% 9 out of 12 (9/12) accordingly. In the 6th trial, the target result which is 100% correct match was attained showing 12 out of 12 (12/12) correct notes.

Testing the Expected Result to Actual Result in the LCD screen

The main feature of the design is to display a chord produced by strumming or plucking certain combinations of notes. This test verifies that feature, where it is the main function of the device. Having these series of tests will show how accurate and reliable the outcome of the design. The results will cater to a credible data since checking is done in numerous ways to accomplish the expected results.

In this case, the kind of test is almost the same with testing the frequency and guitar with pick-up. The difference is that a LCD screen is attached to the output of the microcontroller which is attached to the output of each PLL circuit. The microcontroller is programmed using assembly language. The said program is about the notes to be strummed or plucked processing it to find a match and output a chord to that specific combination. As the strings are struck, signals are then sent to the PLL to filter it according to its corresponding signal. After it is filtered, the signal then goes to the PIC microcontroller having a present or absent state denoting as logic 1 or 0 accordingly. In the microcontroller, the signal is then processed to which chord it complements. The LCD shall then display the chord if it's a match and then informs the user if it's not correct to try again. This is how the testing is tallied in this part. Each trial shall consume 8 seconds as soon as the capture time is started by pressing the button of the LCD screen.

Trial		Time Span	Strum / Pluck	Expected Chord	
		(seconds)	(number of times)	Correct	Not Correct
SET 1	1	8	1	**X**	**√**
	2	8	2	**X**	**√**
	3	8	3	**X**	**√**
	4	8	4	**√**	**X**
	5	8	5	**√**	**X**
SET 2	1	8	1	**X**	**√**
	2	8	2	**√**	**X**
	3	8	3	**√**	**X**
	4	8	4	**X**	**√**
	5	8	5	**√**	**X**
SET 3	1	8	1	**X**	**√**
	2	8	2	**X**	**√**
	3	8	3	**√**	**X**
	4	8	4	**√**	**X**
	5	8	5	**√**	**X**
SET 4	1	8	1	**√**	**X**
	2	8	2	**X**	**√**
	3	8	3	**√**	**X**
	4	8	4	**√**	**X**
	5	8	5	**√**	**X**
SET 5	1	8	1	**X**	**√**
	2	8	2	**√**	**X**
	3	8	3	**√**	**X**
	4	8	4	**√**	**X**
	5	8	5	**√**	**X**
SET 6	1	8	1	**√**	**X**
	2	8	2	**X**	**√**
	3	8	3	**√**	**X**
	4	8	4	**√**	**X**
	5	8	5	**√**	**X**
SET 7	1	8	1	**√**	**X**
	2	8	2	**X**	**√**
	3	8	3	**√**	**X**
	4	8	4	**√**	**X**
	5	8	5	**√**	**X**
SET 8	1	8	1	**X**	**√**
	2	8	2	**√**	**X**
	3	8	3	**√**	**X**
	4	8	4	**√**	**X**
	5	8	5	**√**	**X**

Table 4.3 Guitar with pick-up to LCD displaying the Expected Chord

Each set contains five trials found on Table 4.3 to determine the number of times a chord is strummed or plucked to produce the expected output. A certain trial has a time span of 8 seconds since it is the default capture time of the device. In each set the initial trial will undergo single strumming or plucking of strings and as for the succeeding trials the number of strumming or plucking of strings increments by one. A set is the average result of testing all the chords covered in this study. This means that each chord was checked 8 times (8 sets) to verify its accuracy. From the sets, it was concluded that 28 out of 40 (28/40) giving a 70% rating for expected chord attained. It also gives 12 out of 40 (12/40) giving a 30% rating for expected chord not attained. Some of the factors that may affect these tests results are based on the sensitivity of the guitar pick-up when sound cannot be captured clearly. Another factor is that the notes of the chord played is not tightly pressed down to its fret making the oscillation of the string stop at the instant it is plucked or strummed. Interfering with the oscillation of the string by other means can also affect the performance of the design for it to capture the notes needed. The more the strings are plucked or strummed the more there is a chance to generate the expected output. It is shown in Table 4.3 that 5 out of 8 (5/8) 62.5% sets has a higher success rate of being able to reach the expected chord having three or more strums and plucks, while 3 out of 8 (3/8) or 37.5% will most likely fail if the number of times it is strummed or plucked is 1 or 2 times. The table shows the chord and its corresponding notes as reference in testing the expected results. The reference can be found in Appendix C of this study.

Testing the range covered between the Receiver and Transmitter

A receiver and transmitter may vary its range depending on its type. In this study, an FM receiver and transmitter were used in conducting tests to validate the distance it can cover from one device to another. It is significant to know the maximum range the device can achieve in order for the user to estimate how far he or she may be away from the gadget and still make it work. This kind of test will also determine if other factors may affect the outcome of the output itself. Furthermore, this test showed the limitation of the device so that improvements can be made in the future.

This test was conducted by putting the main board where the receiver was connected to a stable flat surface. This was used as a reference point where the transmitter was connected to the amplifier that was connected to the guitar. The guitar with the transmitter was tested to produce an output at the same point where the receiver was. After one test was conducted, the guitar with the transmitter was taken away from the receiver by 1 meter and was tested again to produce a correct output. This was repeated a number of times until the output on the LCD screen was incorrect or cannot detect any signal from the transmitter. Every checking of the distance is incremented by 1 meter.

Distance **(Meter)**	**Receiver and Transmitter** **(detection of signal)**	**Receiver and Transmitter** **(no detection of signal)**
1	**√**	**X**
2	**√**	**X**
3	**√**	**X**
4	**√**	**X**
5	**√**	**X**
6	**√**	**X**
7	**√**	**X**
8	**√**	**X**
9	**√**	**X**
10	**√**	**X**
11	**X**	**√**
12	**X**	**√**

Table 4.4 Range of Receiver and Transmitter

This table shows the test conducted in determining the maximum distance of detecting a signal. It can be seen that in Table 4.4 a check (**√**) mark can be seen in the column of "detection of signal" from the distance of 1 meter up to the distance of 10 meters indicating that the transmitter and receiver can send and receive signals within the specified range. On the other hand, the column "no detection of signal" has a check (**√**) mark in the distance from 11 to 12 meters indicating that the signal transmitted by the transmitter cannot be detected by the receiver. Through this test, it can be summarized that it can only cover 10m of distance from the transmitter to the receiver. There are also factors that may affect this calculated distance. Some are caused by the thickness of the wall between the two devices; it can also be through other radio signals interference because of such devices.

Chapter 5

CONCLUSION AND RECOMMENDATIONS

Conclusion

A gadget that displayed on a screen the chords made by the user was created. This feat shall help guitar players improve their chord vocabulary as well as remember the chord they are attempting to make. It will also give the user a sense of freedom in doing the desired chord by doing so he can add a certain "feel" to that chord.

The device was made possible by brainstorming as well as conducting rigorous research related to the topic at hand. Its wireless capability is a feature that was attained by using RF (Radio Frequency) Technology. With it, the device can be operational within a certain distance. A series of tests was made to know its limits giving a satisfactory result of a maximum of 10 meters.

It is possible to convert each combination of notes when plucked or strummed from a guitar to a specific chord. These notes can be set as a particular signal which signifies a certain frequency. These signals can be filtered by using the tone decoding principles. PLL (Phase Locked-Loop) technology was also utilized to be able to manifest the signal into its absent or present state to be understood by the microcontroller. Having these ideas put into one, the group was able to produce an expected output which is a chord from the input which are the notes. Again, all of these underwent a series of tests to prove its reliability and accuracy.

Its ease of use is due to its simplicity. A single push button switch was used to start the processing of a guitar chord. The mounting of the display can also be easily understood by the user. Moreover, the steps in using the device are easy to comprehend. A user's manual is also provided to better understand each step.

Recommendations

The designers suggest that not only common chords be deciphered by the device but as well as complicated chords which are seldom used but still are important. It will also be a good idea if the design can be able to display multiple chords at a time so that the previous chords can be stored temporarily for further reference. A water-proof casing can help to protect the circuitry if ever accident spills of liquid happen. For its power supply, an expansion slot wherein a transformer can be used as its source can contribute to saving the battery's lifespan. The display can also be altered to a different output depending on future designing of circuitry so that positions of notes can also be displayed wherein it is called a tablature. It will be more likely practical to improve on the components used, like for example, to use alternative components to improve performance in terms of accuracy of data needed, as well as to minimize the size of parts so that they will be more portable.

BIBLIOGRAPHY

Benson, David J. (2006). Music: A Mathematical Offering, 1st Edition. Cambridge University Press.

Floyd, Thomas L. (2006). Electronic Devices, 6th Edition. Prentice Hall, Colorado.

Julio Sanchez, Maria P. Cantor (2007). Microcontroller Programming: The Microchip PIC

Loy, Gareth. (2006) Musimathics, Volume 1: The Mathematical Foundations of Music. The MIT Press.

APPENDICES

APPENDIX A

Material Listings and Price Lists

Description	Quantity	Unit Price	Sub-Total
LCD module 16 character x 2 Line	1	1,200.00	1,200.00
Mini Push Button	1	10.00	10.00
8 pin connector	2	37.00	74.00
Heatsink	1	20.00	20.00
1200uF/16V electrolytic capacitor	1	6.00	6.00
100uF/25V electrolytic capacitor	1	3.00	3.00
105 multilayer ceramic capacitor	1	2.00	2.00
22pf ceramic capacitor	2	1.00	2.00
W10G Bridge Diode	1	10.00	10.00
1/4Watt resistor	94	0.25	23.50
2 pin terminal block	2	12.00	24.00
4Mhz Crystal	1	50.00	50.00
10K array resistor	2	12.00	24.00
40 pins IC Socket	1	8.00	8.00
PIC16F877 microcontroller IC	1	530.00	530.00
Phone Jack	1	28.00	28.00
Rocker Switch	1	25.00	25.00
LED	37	2.00	74.00
IN4148 Diode	36	2.00	72.00
Trimmer resistor 100K	36	65.00	2,340.00

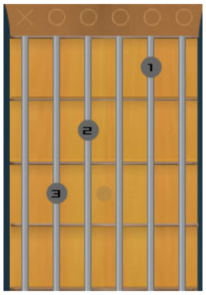
8 pin IC Socket	39	3.00	117.00
104 Multilayer ceramic capacitor	48	2.00	96.00
LM567 IC	36	34.00	1,224.00
LM358 IC	3	34.00	102.00
Alexan Case Black	1	150.00	150.00
Alexan Case White	2	30.00	60.00
Battery Holder	1	45.00	45.00
Battery AA	8	30.00	240.00
10uF/16V electrolytic Capacitor	36	1.00	36.00
47uF/16V electrolytic Capacitor	36	2.00	72.00
Wireless FM Transmitter	1	380.00	380.00
FM Receiver	1	250.00	250.00
Phone Jack Y-adaptor	1	70.00	70.00
Phone Jack converter	1	45.00	45.00
Microphone Amplifier	1	222.00	220.00
9V Battery	1	54.00	54.00
Microphone Cable	1	35.00	35.00
Bag	1	120.00	120.00
TOTAL			7,841.50

APPENDIX B

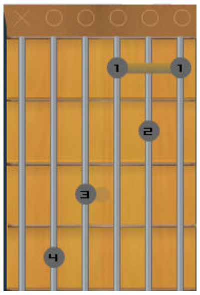
Chord Reference

MAJOR:

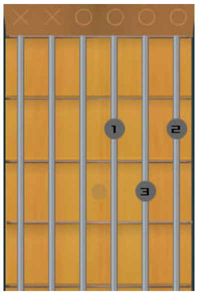
C



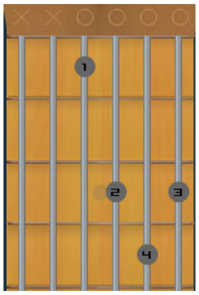
C# / Db



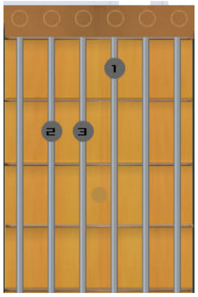
D



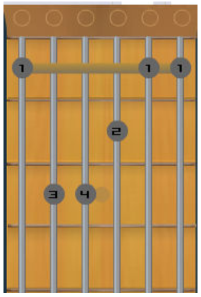
D# / Eb



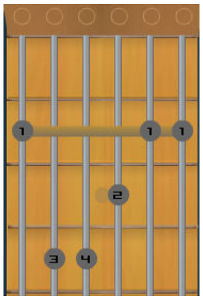
E



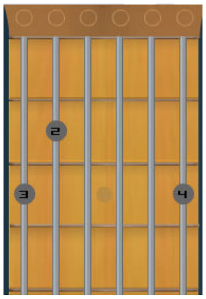
F



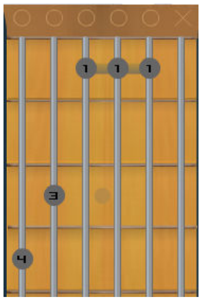
F# / Gb



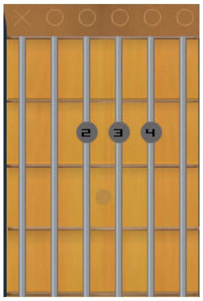
G



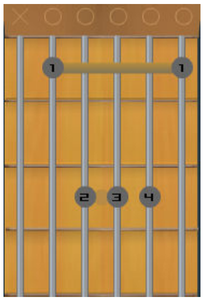
G# / Ab



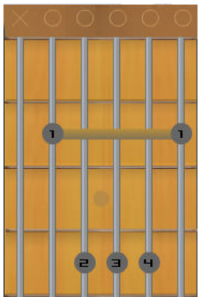
A



A# / Bb

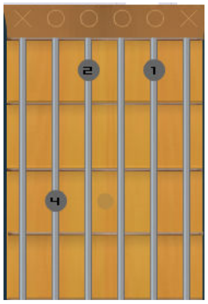


B

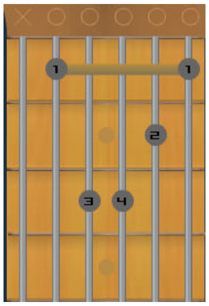


MINOR:

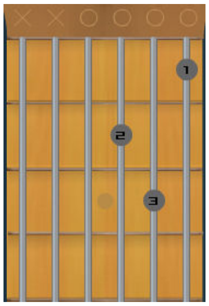
Cm



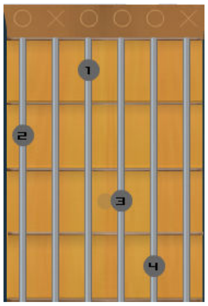
C#m / Dbm (4th Fret)



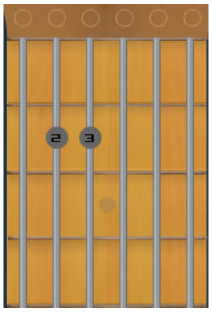
Dm



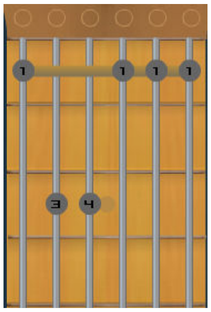
D#m / Ebm



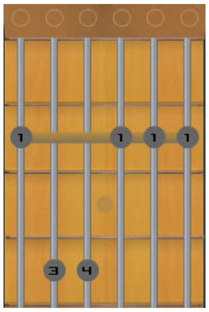
Em



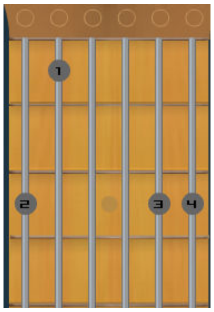
Fm



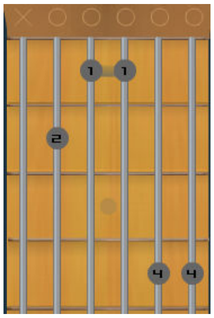
F#m / Gbm



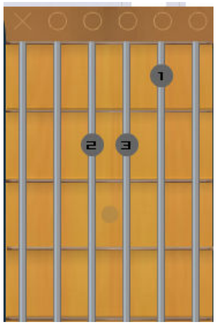
Gm



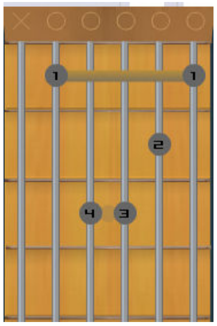
G#m / Abm



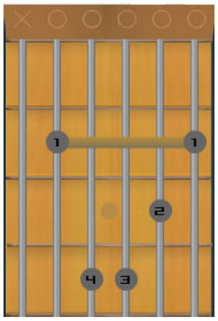
Am



A#m / Bbm

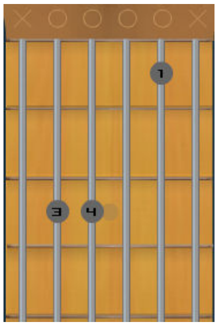


Bm

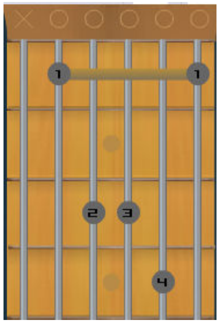


SUSPENDED:

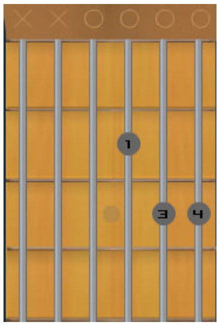
Csus



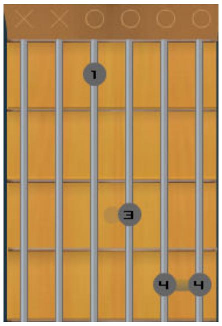
C#sus / Dbsus



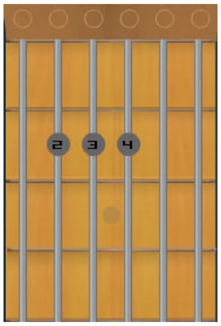
Dsus



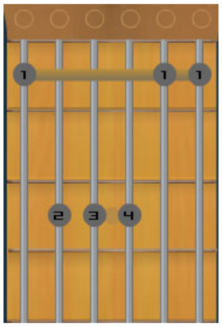
D#sus / Ebsus



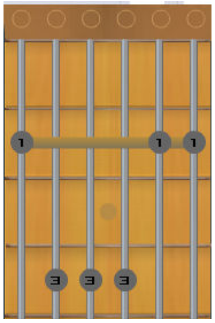
Esus



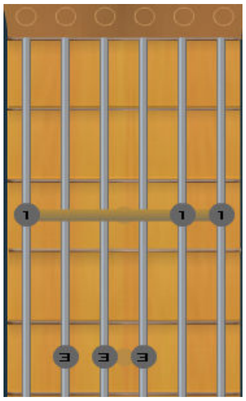
Fsus



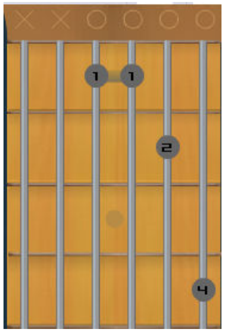
F#sus / Gbsus



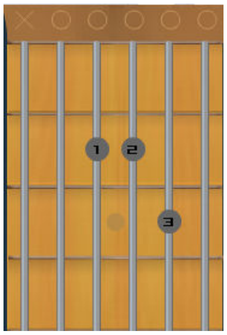
Gsus



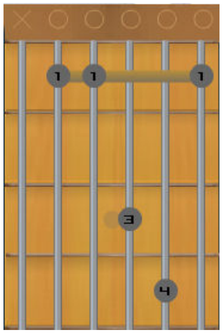
G#sus / Absus



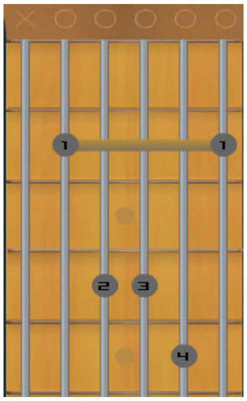
Asus



A#sus / Bbsus

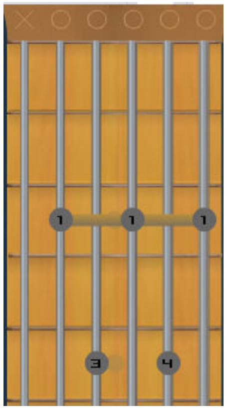


Bsus

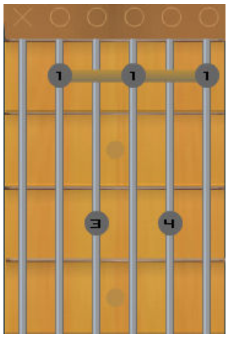


SEVENTH:

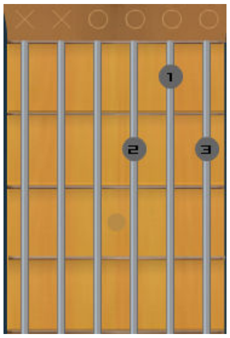
C7



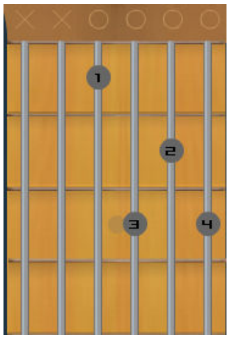
C#7 / Db7 (4th Fret)



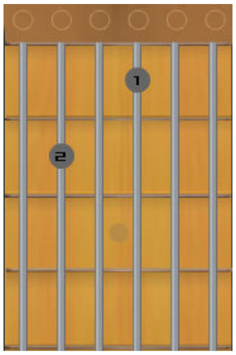
D7



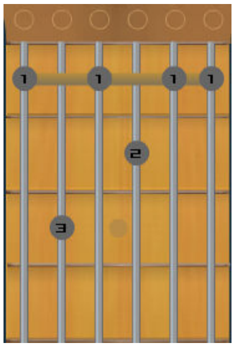
D#7 / Eb7



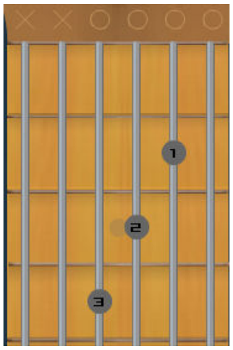
E7



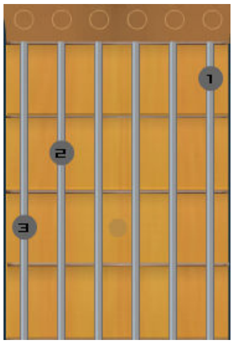
F7



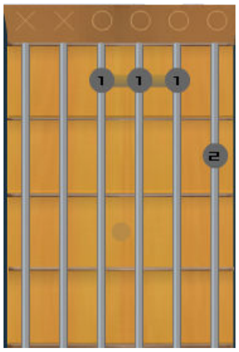
F#7 / Gb7



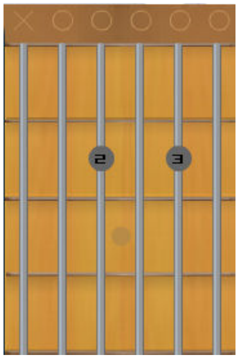
G7



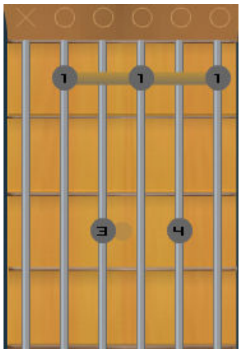
G#7 / Ab7



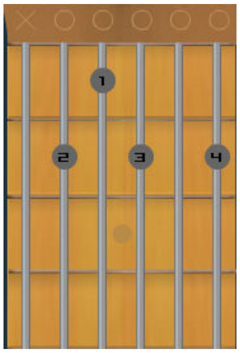
A7



A#7 / Bb7

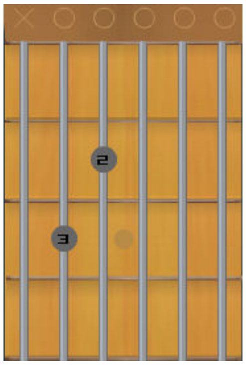


B7

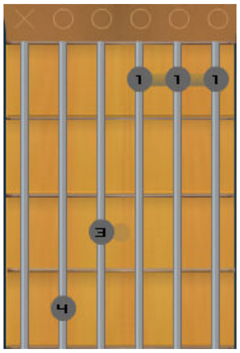


MAJOR(seventh):

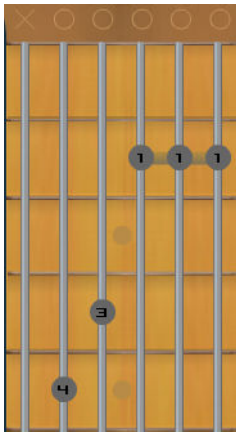
CM7



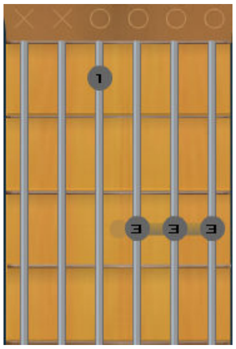
C#M7 / DbM7



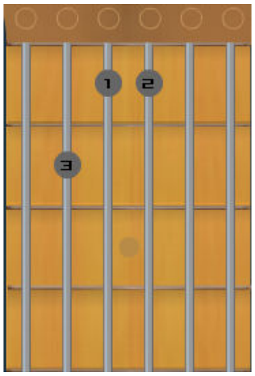
DM7



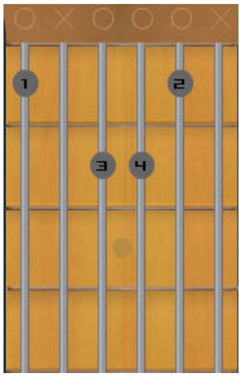
D#M7 / EbM7



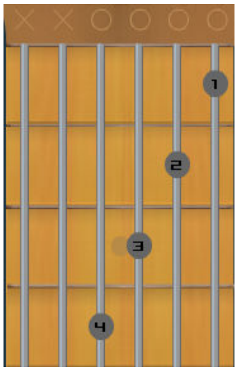
EM7



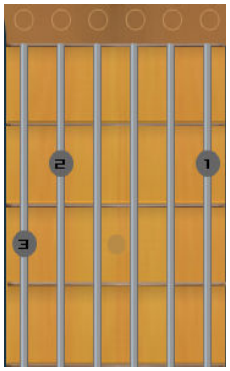
FM7



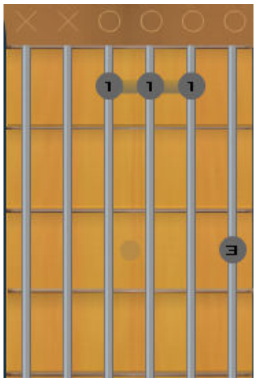
F#M7 / GbM7



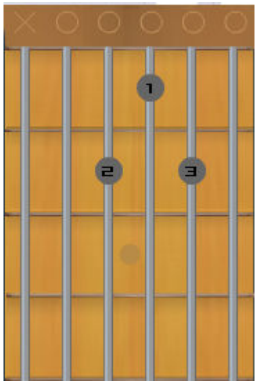
GM7



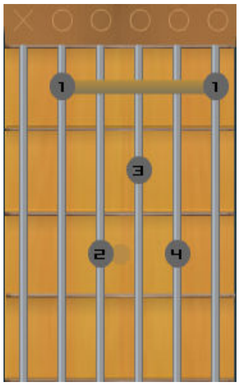
G#M7 / AbM7



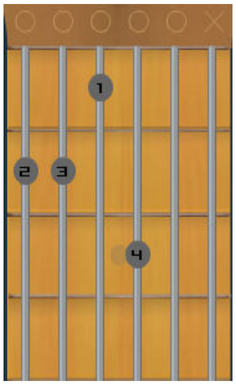
AM7



A#M7 / BbM7

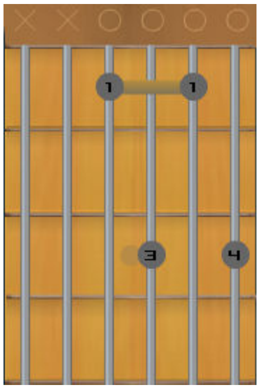


BM7

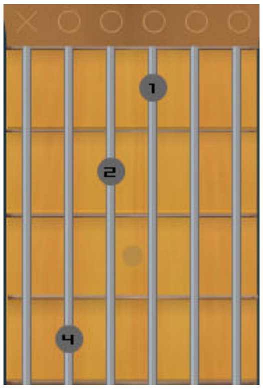


MINOR (seventh):

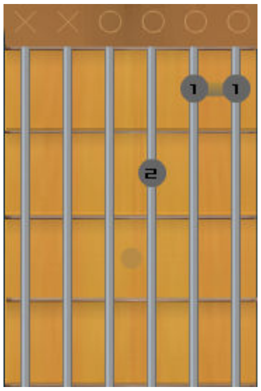
Cm7



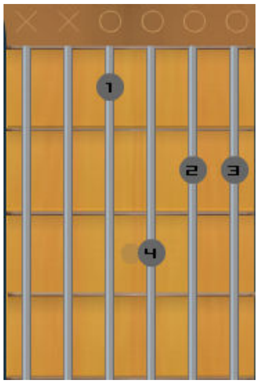
C#m7 / Dbm7



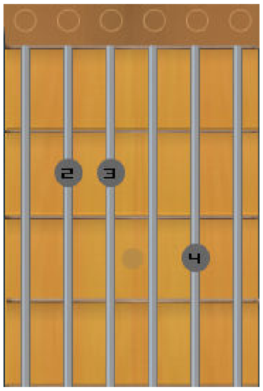
Dm7



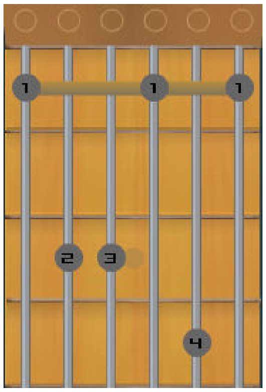
D#m7 / Eb



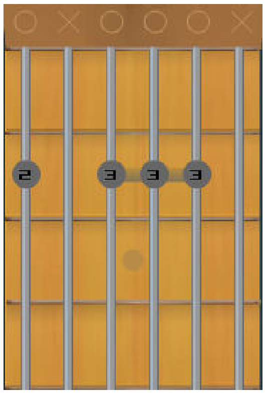
Em7



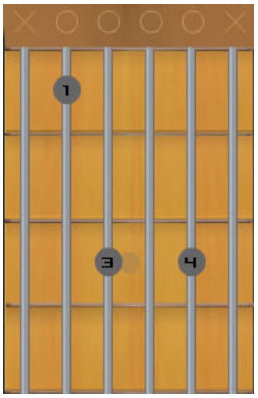
Fm7



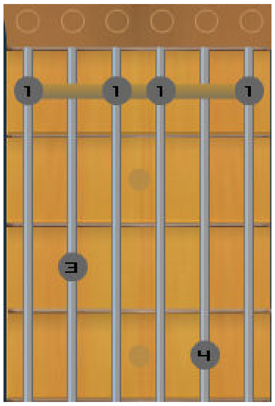
F#m7 / Gbm7



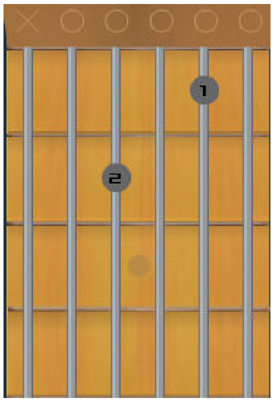
Gm7



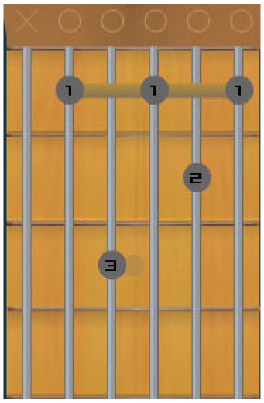
G#m7 / Abm7 (4th Fret)



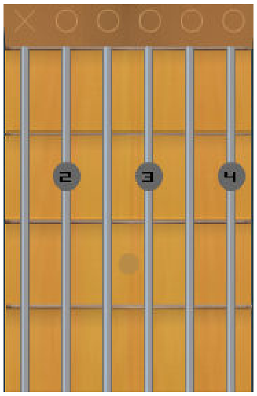
Am7



A#m7 / Bbm7

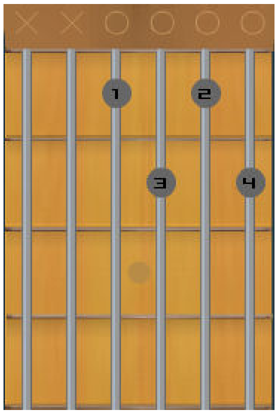


Bm7

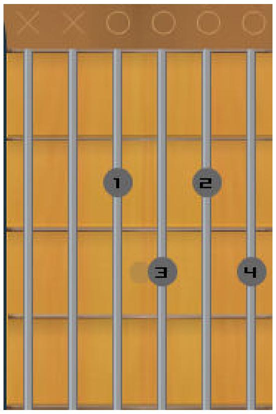


DIMINISHED (dim):

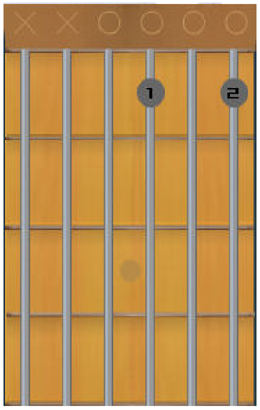
Cdim



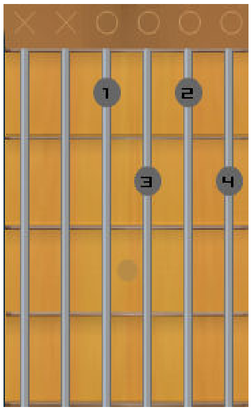
C#dim / Dbdim



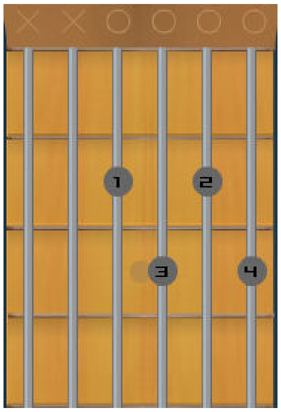
Ddim



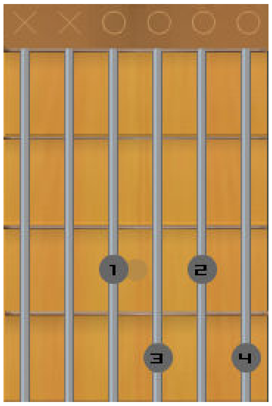
D#dim / Ebdim



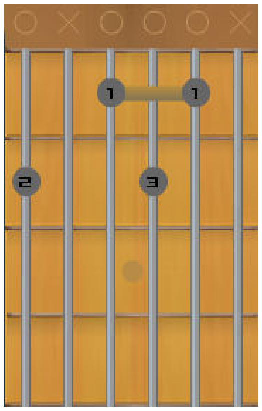
Edim



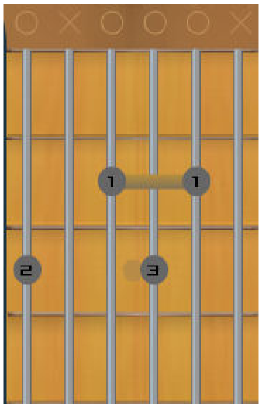
Fdim



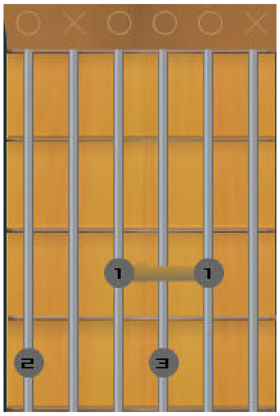
F#dim / Gbdim



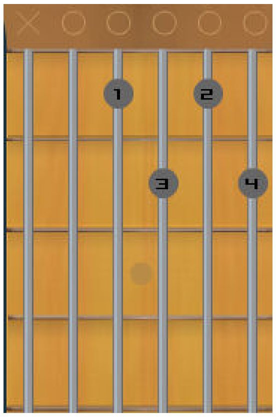
Gdim



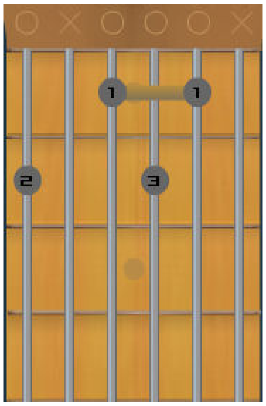
G#dim / Abdim



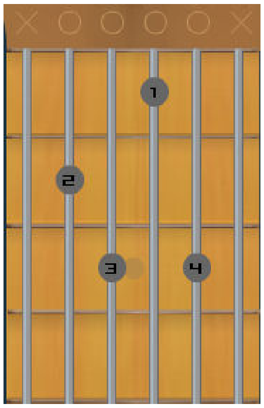
Adim



A#dim / Bbdim (5th Fret)

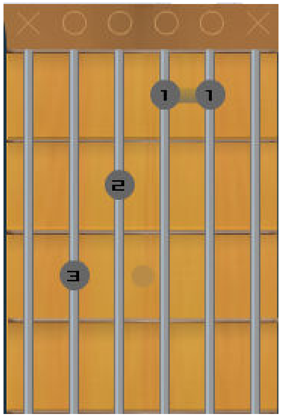


Bdim

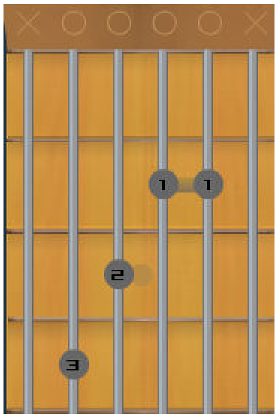


AUGMENTED (aug):

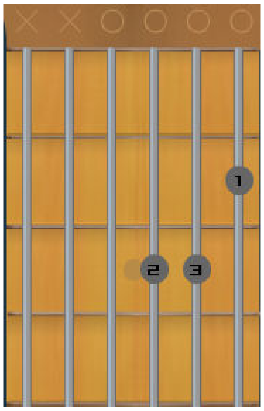
Caug



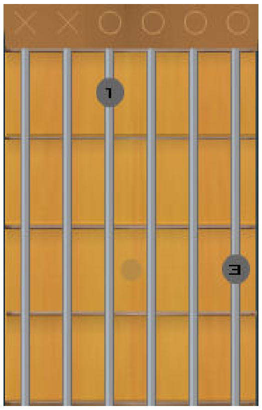
C#aug / Dbaug



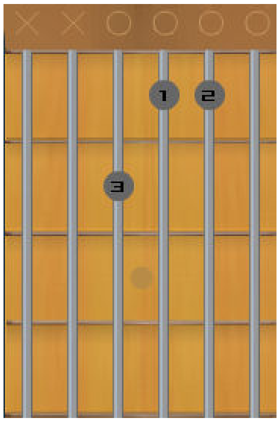
Daug



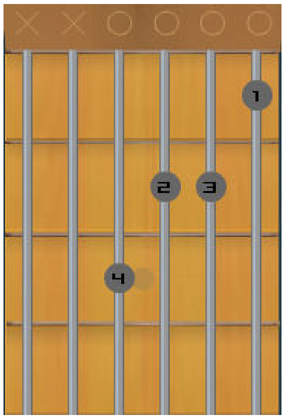
D#aug / Ebaug



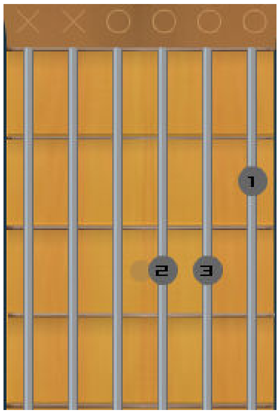
Eaug



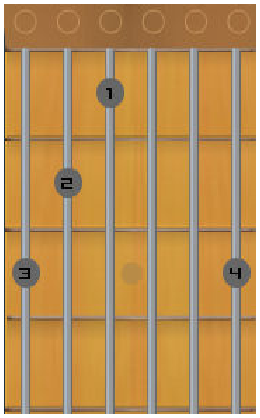
Faug



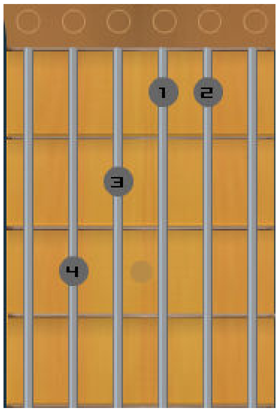
F#aug / Gbaug



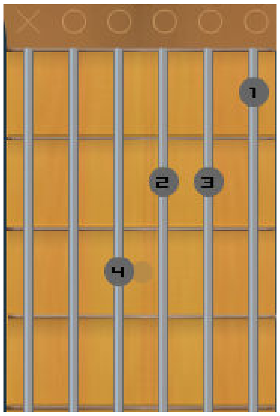
Gaug



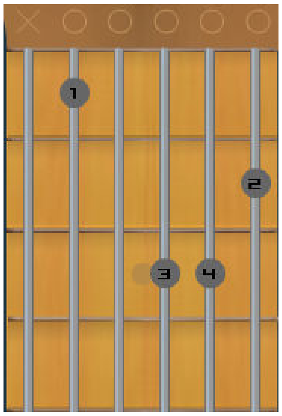
G#aug / Abaug



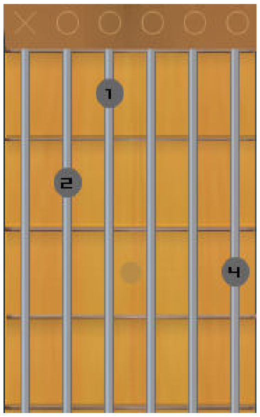
Aaug



A#aug / Bbaug



Baug



APPENDIX C

Chords and its corresponding Notes

Name of Chords **Combination of Notes**

Legend: # -sharp, b - flat

(MAJOR)

C	C	E	G
C# / Db	C# / Db	F	G# / Ab
D	D	F# / Gb	A
D# / Eb	D# / Eb	G	A# / Bb
E	E	G# / Ab	B
F	F	A	C
F# / Gb	F# / Gb	A# / Bb	C# / Db
G	G	B	D
G# / Ab	G# / Ab	C	D# / Eb
A	A	C# / Db	E
A# / Bb	A# / Bb	D	F
B	B	D# / Eb	F# / Gb

Name of Chords **Combination of Notes**

Legend: # -sharp, b - flat

(MINOR)

Cm	C	D# / Eb	G
C#m / Dbm	C# / Db	E	G# / Ab
Dm	D	F	A
D#m / Ebm	D# / Eb	F# / Gb	A# / Bb
Em	E	G	B
Fm	F	G# / Ab	C
F#m / Gbm	F# / Gb	A	C# / Db
Gm	G	A# / Bb	D
G#m / Abm	G# / Ab	B	D# / Eb
Am	A	C	E
A#m / Bbm	A# / Bb	C# / Db	F
Bm	B	D	F# / Gb

Name of Chords **Combination of Notes**

Legend: # -sharp, b - flat

(SUSTAINED / SUSPENDED [sus])

Csus	C	F	G
C#sus / Dbsus	C# / Db	F# / Gb	G# / Ab
Dsus	D	G	A
D#sus / Ebsus	D# / Eb	G# / Ab	A# / Bb
Esus	E	A	B
Fsus	F	A# / Bb	C
F#sus / Gbsus	F# / Gb	B	C# / Db
Gsus	G	C	D
G#sus / Absus	G# / Ab	C# / Db	D# / Eb
Asus	A	D	E
A#sus / Bbsus	A# / Bb	D# / Eb	F
Bsus	B	E	F# / Gb

Name of Chords **Combination of Notes**

Legend: # -sharp, b - flat

(SEVENTH [7th])

C7	C	E	G	A# / Bb
C#7 / Db7	C# / Db	F	G# / Ab	B
D7	D	F# / Gb	A	C
D#7 / Eb7	D# / Eb	G	A# / Bb	C# / Db
E7	E	G# / Ab	B	D
F7	F	A	C	D# / Eb
F#7 / Gb7	F# / Gb	A# / Bb	C# / Db	E
G7	G	B	D	F
G#7 / Ab7	G# / Ab	C	D# / Eb	F# / Gb
A7	A	C# / Db	E	G
A#7 / Bb7	A# / Bb	D	F	G# / Ab
B7	B	D# / Eb	F# / Gb	A

Name of Chords **Combination of Notes**

Legend: # -sharp, b - flat

(MAJOR [7th])

CM7	C	E	G	B
C#M7 / DbM7	C# / Db	F	G# / Ab	C
DM7	D	F# / Gb	A	C# / Db
D#M7 / EbM7	D# / Eb	G	A# / Bb	D
EM7	E	G# / Ab	B	D# / Eb
FM7	F	A	C	E
F#M7 / GbM7	F# / Gb	A# / Bb	C# / Db	F
GM7	G	B	D	F# / Gb
G#M7 / AbM7	G# / Ab	C	D# / Eb	G
AM7	A	C# / Db	E	G# / Ab
A#M7 / BbM7	A# / Bb	D	F	A
BM7	B	D# / Eb	F# / Gb	A# / Bb

Name of Chords **Combination of Notes**

Legend: # -sharp, b - flat

(MINOR [7th])

Cm7	C	D# / Eb	G	A# / Bb
C#m7 / Dbm7	C# / Db	E	G# / Ab	B
Dm7	D	F	A	C
D#m7 / Ebm7	D# / Eb	F# / Gb	A# / Bb	C# / Db
Em7	E	G	B	D
Fm7	F	G# / Ab	C	D# / Eb
F#m7 / Gbm7	F# / Gb	A	C# / Db	E
Gm7	G	A# / Bb	D	F
G#m7 / Abm7	G# / Ab	B	D# / Eb	F# / Gb
Am7	A	C	E	G
A#m7 / Bbm7	A# / Bb	C# / Db	F	G# / Ab
Bm7	B	D	F# / Gb	A

Name of Chords **Combination of Notes**

Legend: # -sharp, b - flat

(DIMINISHED [dim])

Cdim	C	D# / Eb	F# / Gb	A
C#dim / Dbdim	C# / Db	E	G	A# / Bb
Ddim	D	F	G# / Ab	B
D#dim / Ebdim	D# / Eb	F# / Gb	A	C
Edim	E	G	A# / Bb	C# / Db
Fdim	F	G# / Ab	B	D
F#dim / Gbdim	F# / Gb	A	C	D# / Eb
Gdim	G	A# / Bb	C# / Db	E
G#dim / Abdim	G# / Ab	B	D	F
Adim	A	C	D# / Eb	F# / Gb
A#dim / Bbdim	A# / Bb	C# / Db	E	G
Bdim	B	D	F	G# / Ab

Name of Chords **Combination of Notes**

Legend: # -sharp, b - flat

(AUGMENTED [aug])

Caug	C	E	G# / Ab
C#aug / Dbaug	C# / Db	F	A
Daug	D	F# / Gb	A# / Bb
D#aug / Ebaug	D# / Eb	G	B
Eaug	E	G# / Ab	C
Faug	F	A	C# / Db
F#aug / Gbaug	F# / Gb	A# / Bb	D
Gaug	G	B	D# / Eb
G#aug / Abaug	G# / Ab	C	E
Aaug	A	C# / Db	F
A#aug / Bbaug	A# / Bb	D	F# / Gb
Baug	B	D# / Eb	G

APPENDIX D

LM 567 IC Tone Decoder

National Semiconductor

February 1995

LM567/LM567C Tone Decoder

General Description

The LM567 and LM567C are general purpose tone decoders designed to provide a saturated transistor switch to ground when an input signal is present within the passband. The circuit consists of an I and Q detector driven by a voltage controlled oscillator which determines the center frequency of the decoder. External components are used to independently set center frequency, bandwidth and output delay.

Features

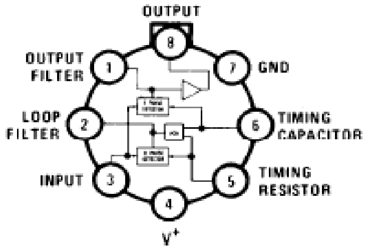
- 20 to 1 frequency range with an external resistor
- Logic compatible output with 100 mA current sinking capability
- Bandwidth adjustable from 0 to 14%
- High rejection of out of band signals and noise
- Immunity to false signals
- Highly stable center frequency
- Center frequency adjustable from 0.01 Hz to 500 kHz

Applications

- Touch tone decoding
- Precision oscillator
- Frequency monitoring and control
- Wide band FSK demodulation
- Ultrasonic controls
- Carrier current remote controls
- Communications paging decoders

Connection Diagrams

Metal Can Package

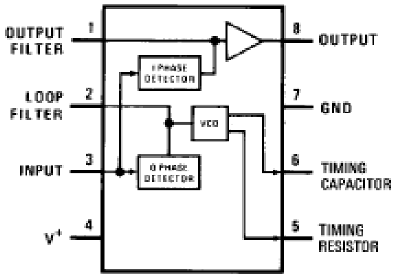


TL/H/6975–1

Top View

Order Number LM567H or LM567CH
See NS Package Number H08C

Dual-In-Line and Small Outline Packages



TL/H/6975–2

Top View

Order Number LM567CM
See NS Package Number M08A
Order Number LM567CN
See NS Package Number N08E

TL/H/6975

RRD-B30M115/Printed in U. S. A.

APPENDIX E

PIC16F877 Microcontroller IC

MICROCHIP

PIC16F87X

28/40-Pin 8-Bit CMOS FLASH Microcontrollers

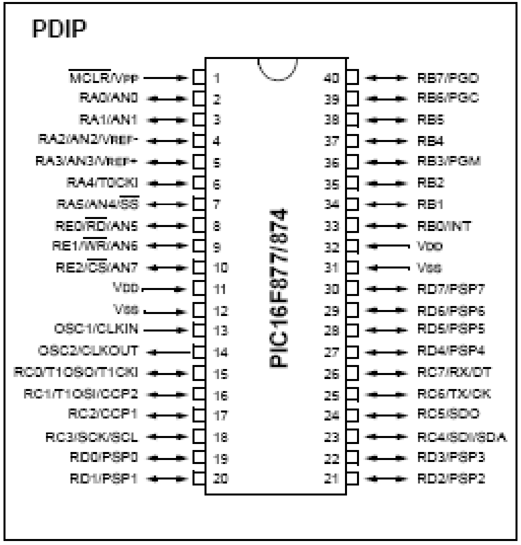
Devices Included in this Data Sheet:

- PIC16F873
- PIC16F874
- PIC16F876
- PIC16F877

Microcontroller Core Features:

- High performance RISC CPU
- Only 35 single word instructions to learn
- All single cycle instructions except for program branches which are two cycle
- Operating speed: DC - 20 MHz clock input
  DC - 200 ns instruction cycle
- Up to 8K x 14 words of FLASH Program Memory,
  Up to 368 x 8 bytes of Data Memory (RAM)
  Up to 256 x 8 bytes of EEPROM Data Memory
- Pinout compatible to the PIC16C73B/74B/76/77
- Interrupt capability (up to 14 sources)
- Eight level deep hardware stack
- Direct, indirect and relative addressing modes
- Power-on Reset (POR)
- Power-up Timer (PWRT) and Oscillator Start-up Timer (OST)
- Watchdog Timer (WDT) with its own on-chip RC oscillator for reliable operation
- Programmable code protection
- Power saving SLEEP mode
- Selectable oscillator options
- Low power, high speed CMOS FLASH/EEPROM technology
- Fully static design
- In-Circuit Serial Programming™ (ICSP) via two pins
- Single 5V In-Circuit Serial Programming capability
- In-Circuit Debugging via two pins
- Processor read/write access to program memory
- Wide operating voltage range: 2.0V to 5.5V
- High Sink/Source Current: 25 mA
- Commercial, Industrial and Extended temperature ranges
- Low-power consumption:
  - < 0.6 mA typical @ 3V, 4 MHz
  - 20 µA typical @ 3V, 32 kHz
  - < 1 µA typical standby current

Pin Diagram



Peripheral Features:

- Timer0: 8-bit timer/counter with 8-bit prescaler
- Timer1: 16-bit timer/counter with prescaler, can be incremented during SLEEP via external crystal/clock
- Timer2: 8-bit timer/counter with 8-bit period register, prescaler and postscaler
- Two Capture, Compare, PWM modules
  - Capture is 16-bit, max. resolution is 12.5 ns
  - Compare is 16-bit, max. resolution is 200 ns
  - PWM max. resolution is 10-bit
- 10-bit multi-channel Analog-to-Digital converter
- Synchronous Serial Port (SSP) with SPI™ (Master mode) and I^2C™ (Master/Slave)
- Universal Synchronous Asynchronous Receiver Transmitter (USART/SCI) with 9-bit address detection
- Parallel Slave Port (PSP) 8-bits wide, with external $\overline{RD}$, $\overline{WR}$ and $\overline{CS}$ controls (40/44-pin only)
- Brown-out detection circuitry for Brown-out Reset (BOR)

APPENDIX F

Phase-Locked Loop

National Semiconductor

November 1995

CD4046BM/CD4046BC Micropower Phase-Locked Loop

General Description

The CD4046B micropower phase-locked loop (PLL) consists of a low power, linear, voltage-controlled oscillator (VCO), a source follower, a zener diode, and two phase comparators. The two phase comparators have a common signal input and a common comparator input. The signal input can be directly coupled for a large voltage signal, or capacitively coupled to the self-biasing amplifier at the signal input for a small voltage signal.

Phase comparator I, an exclusive OR gate, provides a digital error signal (phase comp. I Out) and maintains 90° phase shifts at the VCO center frequency. Between signal input and comparator input (both at 50% duty cycle), it may lock onto the signal input frequencies that are close to harmonics of the VCO center frequency.

Phase comparator II is an edge-controlled digital memory network. It provides a digital error signal (phase comp. II Out) and lock-in signal (phase pulses) to indicate a locked condition and maintains a 0° phase shift between signal input and comparator input.

The linear voltage-controlled oscillator (VCO) produces an output signal (VCO Out) whose frequency is determined by the voltage at the VCO_{IN} input, and the capacitor and resistors connected to pin $C1_A$, $C1_B$, R1 and R2.

The source follower output of the VCO_{IN} (demodulator Out) is used with an external resistor of 10 kΩ or more.

The INHIBIT input, when high, disables the VCO and source follower to minimize standby power consumption. The zener diode is provided for power supply regulation, if necessary.

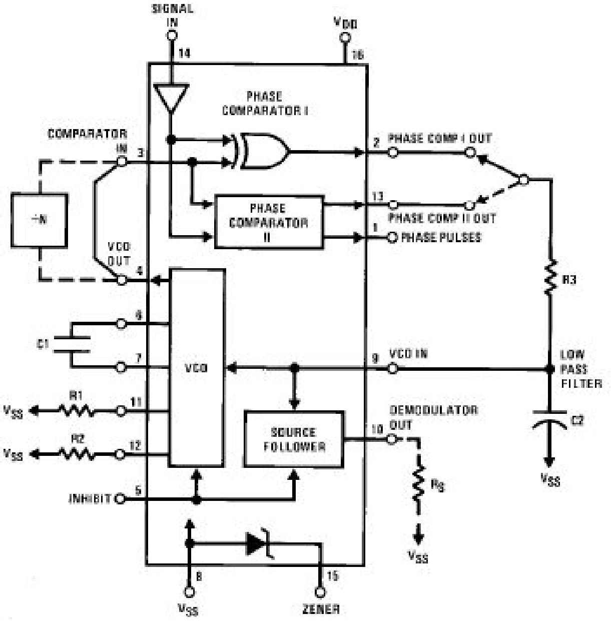
Features

■ Wide supply voltage range	3.0V to 18V
■ Low dynamic power consumption	70 μW (typ.) at f_o = 10 kHz, V_{DD} = 5V
■ VCO frequency	1.3 MHz (typ.) at V_{DD} = 10V
■ Low frequency drift with temperature	0.06%/°C at V_{DD} = 10V
■ High VCO linearity	1% (typ.)

Applications

- FM demodulator and modulator
- Frequency synthesis and multiplication
- Frequency discrimination
- Data synchronization and conditioning
- Voltage-to-frequency conversion
- Tone decoding
- FSK modulation
- Motor speed control

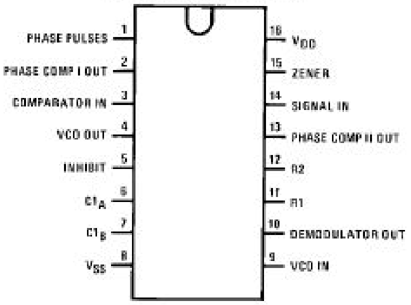
Block & Connection Diagrams



TL/F/5968–1

FIGURE 1

Dual-In-Line Package



TL/F/5968–2

Top View

Order Number CD4046B

TL/F/5968

APPENDIX G

LCD Module

SC1602D (16 CHARACTERS x 2 LINES)

FEATURES

- 5 x 7 DOTS WITH CURSOR
- BUILT-IN CONTROLLER (KS0066 OR EQUIVALENT)
- 5 V POWER SUPPLY
- 1/16 DUTY CYCLE
- 4.2 V LED FORWARD VOLTAGE

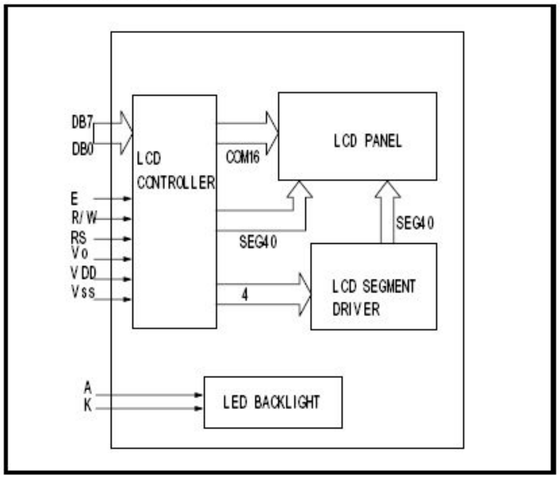
ELECTRICAL CHARACTERISTICS

ITEM		SYMBOL	CONDITION	MIN.	TYP.	MAX.	UNIT
LCD Operating Voltage		V_{DD}-Vo	T=0 °C	-	4.8	-	V
			T=25 °C	-	4.5	-	V
			T=50 °C	-	4.2	-	V
Supply Voltage		V_{DD}-Vss	-	4.7	5	5.3	V
Supply Current		I_{DD}	-	-	2	4	mA
Input Voltage	"HIGH" Level	V_{IH}	-	2.2	-	V_{DD}	V
	"LOW" Level	V_{IL}	-	0	-	0.6	V
Output Voltage	"HIGH" Level	V_{OH}	-	2.4	-	-	V
	"LOW" Level	V_{OL}	-	-	-	0.4	V

MECHANICAL DATA

ITEM	DIMENSIONS	UNIT
Module Size (W x H x T)	85.0 x 36.0 x 8.8 (12.7 LED)	mm
Viewing Area (W x H)	65.0 x 16.0	mm
Character Size (W x H)	2.96 x 5.56	mm
Character Pitch (W x H)	3.55 x 5.94	mm
Dot Size (W x H)	0.56 x 0.66	mm
Dot Pitch (W x H)	0.60 x 0.70	mm

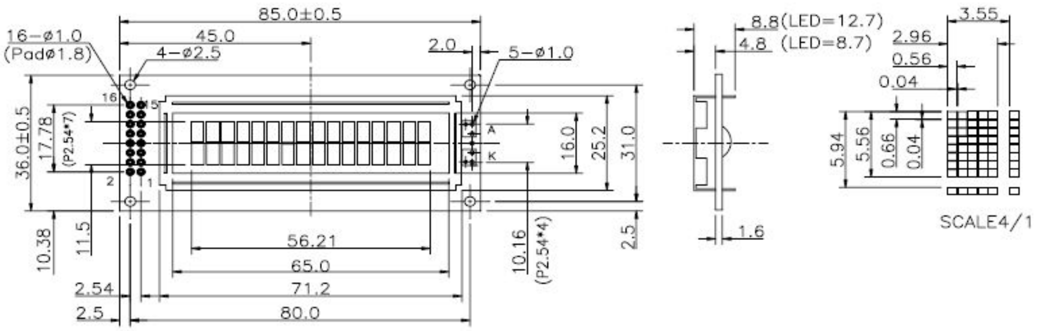
BLOCK DIAGRAM



INTERFACE PIN CONNECTIONS

NO.	SYMBOL	FUNCTION	NO.	SYMBOL	FUNCTION
1	Vss	Supply Ground	9	DB2	Data Bit 2
2	V_{DD}	Supply Voltage	10	DB3	Data Bit 3
3	Vo	Contrast Adj.	11	DB4	Data Bit 4
4	RS	Register Select	12	DB5	Data Bit 5
5	R/W	Read/Write	13	DB6	Data Bit 6
6	E	Enable Signal	14	DB7	Data Bit 7
7	DB0	Data Bit 0	15	A	LED Power
8	DB1	Data Bit 1	16	K	LED Power

EXTERNAL DIMENSIONS



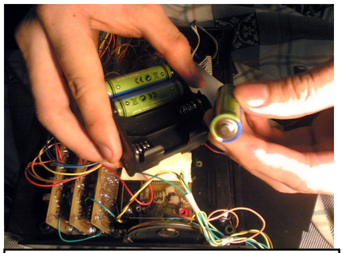
APPENDIX H

User's Manual

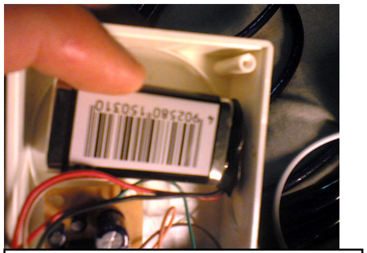
How to use the Wireless Chord Creator for Guitars with Pick-Ups:

Set-Up:

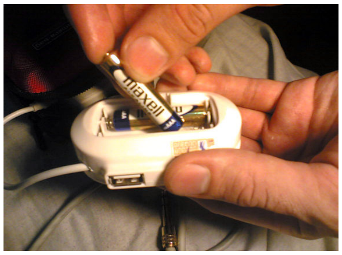
1. Put batteries on the following; 8pcs. AA Batteries [1.5V] in main unit [Black Box], 1pc. 9V Battery in amplifier [White Box], and 2pcs. AAA Batteries [1.5V] FM transmitter [White Case].



Main unit [Black Box]

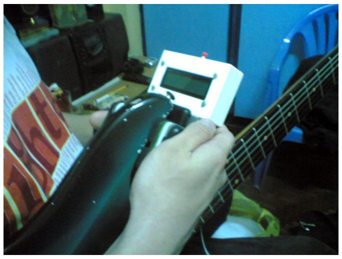


Amplifier [White Box]



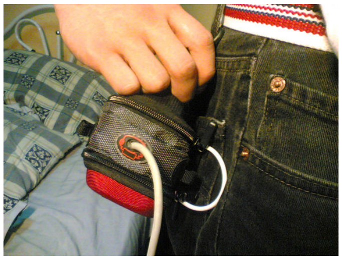
FM Transmitter [White Case]

2. Loosen the clip on the LCD display to be able to clamp it on the preferred part of the guitar or place it at a convenient area that the user will be able to see the output clearly.

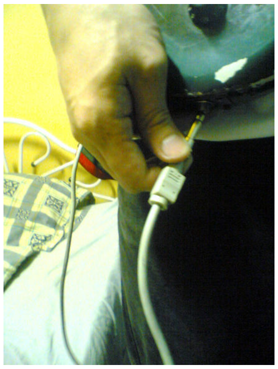


3. Place the main unit [Black Box] on the floor or any area that you may prefer close enough to your guitar not to pull the cable attached to it too much. Switch the power ‘On’.

4. Place the Amplifier [White Box] and FM Transmitter [White Case] to its proper casing [Small Bag(black and red) with clip]. It is highly suggested to clip the bag on your pants.



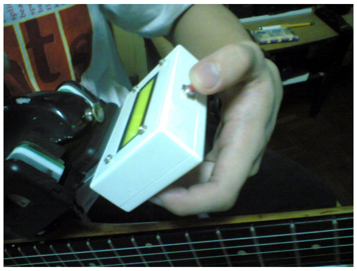
5. Plug the cable from the transmitter to the guitar to be used.

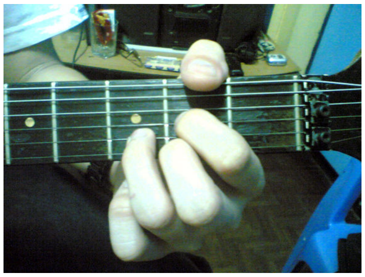


6. Switch-On the power for the FM transmitter unit by holding down the power button for 3 sec. and set the default frequency to 107.1MHz by pressing the ‘+’ or ‘-’ sign. Also switch-on the amplifier.

Usage:

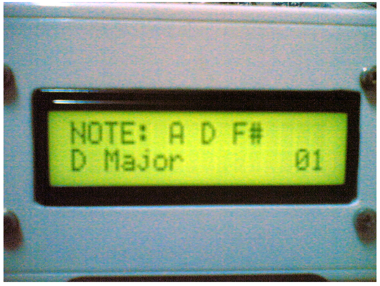
1. Push the Red Button of the LCD display to start the capture time (8 seconds capture time) and then do a chord on the frets (guitar neck / fretboard) or vice-versa.



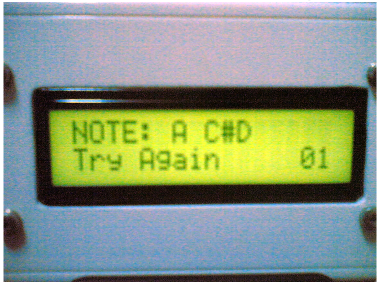


2. Strum or pluck the guitar strings while timer is still counting until it reaches 1 millisecond.

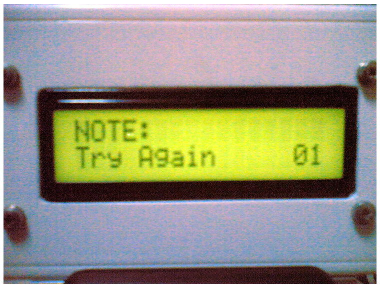
3. During the capture time the notes that are plucked or strung will be displayed. After the capture time if the combination of notes corresponds to the correct combination, the Chord will be displayed, if not or there are no strings strung or plucked the display is "Try Again".



```
NOTE: A D F#
D Major       01
```



```
NOTE: A C#D
Try Again     01
```

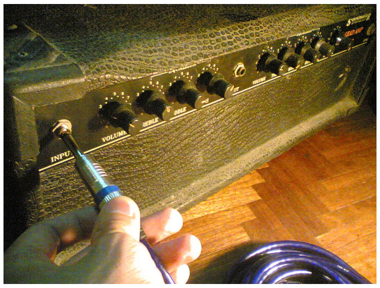


```
NOTE:
Try Again     01
```

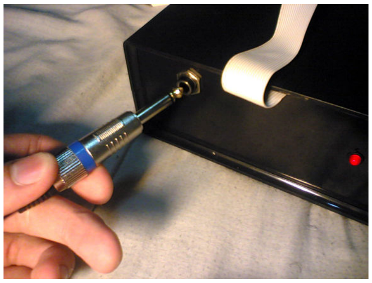
4. Repeat from step 1 in Usage if you want to try other chords.
5. If done using, shut-off the units to preserve the battery lifespan.

Optional:

- If there is a need to plug-in the guitar to an Amplifier System, simply plug a cable on the extra slot for the output of the Transmitter package to the Amplifier. Plug the other end of the cable to the Amplifier System.



- If you want to directly plug into the main unit [Black Box] from the guitar, you can do so. Just plug-in one end of the cable to the guitar and the other end to the main unit. (If this is the approach, the small bag (black and red) consisting of the amplifier and transmitter is not used.)



APPENDIX I

Source Code

```
;Variable Declaration
PortA_New     equu   H'20'
PortC_New     equu   H'21'
PortE_New     equu   H'22'
PortE_Prev    equ   H'23'

Note_Lo       equ   H'28'
Note_Hi       equ   H'29'

Tmr1_Sec      equ   H'30'
Tmr1_Pres     equ   H'31'

Wait1_Val     equ   H'71'
Wait2_Val     equ   H'72'
Msg_Num       equ   H'73'

Temp1         equ   H'79'
Temp2         equ   H'7A'
Temp3         equ   H'7B'
Temp4         equ   H'7C'
W_TEMP        equ   H'7D'
STAT_TEMP     equ   H'7E'
PCLATH_TEMP   equ   H'7F'

LCD_RAM_Buf   equ   H'20'

;Reset Vector Starts at Address 0x0000.
      org   0x0000
      goto  Initialize

      org   0x0004
      goto  ISR_routine

;Initialization Routine.
Initialize:clrf  TMR0
         clrf  INTCON
         bcf   STATUS,RP1
         bsf   STATUS,RP0
         movlw B'11000011'
         movwf OPTION_REG

         movlw B'00000110'
         movwf ADCON1
         movlw B'11111111'
         movwf TRISA

         movlw B'00000000'
         movwf TRISB

         movlw B'11111111'
         movwf TRISC

         movlw B'00000000'
         movwf TRISD

         movlw B'00000111'
         movwf TRISE

         bcf   STATUS,RP0

         call  Init_Var
         call  Init_LCD
         call  Disp_LCD

         bsf   INTCON,T0IE
         bsf   INTCON,GIE

;Main Program Starts Here.
Main:    nop
         goto  Main

;The Interrupt Service Routine.
ISR_routine:
         movwf W_TEMP
         movf  STATUS,W
         movwf STAT_TEMP
         bcf   STATUS,RP0

         btfsc INTCON,T0IF
         goto  TMR0int

RestoreReg:
         movf  STAT_TEMP,W
         movwf STATUS
         movf  W_TEMP,W

         retfie

;TIMER 0 (TMR0)Interrupt Service Routine.
TMR0int:  bcf   INTCON,T0IF
         movlw D'06'
         addwf TMR0,F

         call  Read_Input
         call  Do_Tmr1
         call  Disp_Data
         call  Disp_Chord
         call  Disp_LCD

TMR0intX:  goto  RestoreReg

Msg0:    addwf PCL,F

         dt   "NOTE:           "
         dt   "                "
```

```
Init_Var:clrf  Msg_Num
         call  Ld_Msg2RAM
         movf  PORTA,W
         movwf PortA_New
         movf  PORTE,W
         movwf PortE_New
         movwf PortE_Prev
         movf  PORTC,W
         movwf PortC_New
         clrf  PORTD
         clrf  Tmr1_Sec
         clrf  Tmr1_Pres
         clrf  Note_Lo
         clrf  Note_Hi
         return

Read_Input:movf  PORTA,W
           movwf PortA_New
           movf  PORTC,W
           movwf PortC_New
           movf  PORTE,W
           movwf PortE_New

Read_RE2:btfsc PortE_New,2
         goto  Read_RE2X
         btfss PortE_Prev,2
         goto  Read_RE2X
         movf  Tmr1_Sec,W
         btfss STATUS,Z
         goto  Read_RE2X
         movlw H'80'
         movwf Tmr1_Sec
         movlw D'0'
         call  Ld_Msg2RAM
         clrf  Note_Lo
         clrf  Note_Hi
Read_RE2X:nop

Chk_Tmr1:movf  Tmr1_Sec,W
         btfsc STATUS,Z
         goto  Chk_Tmr1X

Read_RA0:btfss PortA_New,0
         bsf   Note_Lo,0

Read_RA1:btfss PortA_New,1
         bsf   Note_Lo,1

Read_RA2:btfss PortA_New,2
         bsf   Note_Lo,2

Read_RA3:btfss PortA_New,3
         bsf   Note_Lo,3
```

```
Read_RA4:btfss PortA_New,4
         bsf   Note_Lo,4

Read_RA5:btfss PortA_New,5
         bsf   Note_Lo,5

Read_RC0:btfss PortC_New,0
         bsf   Note_Hi,0

Read_RC1:btfss PortC_New,1
         bsf   Note_Hi,1

Read_RC2:btfss PortC_New,2
         bsf   Note_Hi,2

Read_RC3:btfss PortC_New,3
         bsf   Note_Hi,3

Read_RC4:btfss PortC_New,4
         bsf   Note_Hi,4

Read_RC5:btfss PortC_New,5
         bsf   Note_Hi,5

Chk_Tmr1X:nop

Read_InX:movf  PortE_New,W
         movwf PortE_Prev
         return

Disp_Data:movf   Tmr1_Sec,W
          btfsc  STATUS,Z
          goto   Disp_DataX

          movlw  LCD_RAM_Buf
          addlw  D'6'
          movwf  FSR
          bsf    FSR,7

DispA:    btfss  Note_Lo,0
          goto   DispAX
          movlw  "A"
          movwf  INDF
          incf   FSR,F
          movlw  " "
          movwf  INDF
          incf   FSR,F
DispAX:   nop

DispBb:   btfss  Note_Lo,1
          goto   DispBbX
          movlw  "B"
          movwf  INDF
          incf   FSR,F
          movlw  "b"
          movwf  INDF
          incf   FSR,F
DispBbX: nop

DispB:    btfss  Note_Lo,2
          goto   DispBX
          movlw  "B"
          movwf  INDF
          incf   FSR,F
          movlw  " "
          movwf  INDF
          incf   FSR,F
DispBX:   nop

DispC:    btfss  Note_Lo,3
          goto   DispCX
          movlw  "C"
          movwf  INDF
          incf   FSR,F
          movlw  " "
          movwf  INDF
          incf   FSR,F
DispCX:    nop

DispCb:   btfss  Note_Lo,4
          goto   DispCbX
          movlw  "C"
          movwf  INDF
          incf   FSR,F
          movlw  "#"
          movwf  INDF
          incf   FSR,F
DispCbX: nop

DispD:     btfss  Note_Lo,5
          goto   DispDX
          movlw  "D"
          movwf  INDF
          incf   FSR,F
          movlw  " "
          movwf  INDF
          incf   FSR,F
DispDX:   nop

DispDb:   btfss  Note_Hi,0
          goto   DispDbX
          movlw  "D"
          movwf  INDF
          incf   FSR,F
          movlw  "#"
          movwf  INDF
```

```
          incf   FSR,F
DispDbX: nop

DispE:    btfss  Note_Hi,1
          goto   DispEX
          movlw  "E"
          movwf  INDF
          incf   FSR,F
          movlw  " "
          movwf  INDF
          incf   FSR,F
DispEX:   nop

DispF:    btfss  Note_Hi,2
          goto   DispFX
          movlw  "F"
          movwf  INDF
          incf   FSR,F
          movlw  " "
          movwf  INDF
          incf   FSR,F
DispFX:   nop

DispFb:    btfss  Note_Hi,3
          goto   DispFbX
          movlw  "F"
          movwf  INDF
          incf   FSR,F
          movlw  "#"
          movwf  INDF
          incf   FSR,F
DispFbX:  nop

DispG:   btfss  Note_Hi,4
          goto   DispGX
          movlw  "G"
          movwf  INDF
          incf   FSR,F
          movlw  " "
          movwf  INDF
          incf   FSR,F
DispGX:  nop

DispGb:   btfss  Note_Hi,5
          goto   DispGbX
          movlw  "G"
          movwf  INDF
          incf   FSR,F
          movlw  "#"
          movwf  INDF
          incf   FSR,F
DispGbX: nop
```

```
            movlw  LCD_RAM_Buf
            addlw  D'30'
            movwf  FSR
            bsf    FSR,7

            swapf  Tmr1_Sec,W
            andlw  H'0F'
            addlw  H'30'
            movwf  INDF
            incf   FSR,F
            movf   Tmr1_Sec,W
            andlw  H'0F'
            addlw  H'30'
            movwf  INDF

Disp_DataX:return

Do_Tmr1:movf   Tmr1_Sec,W
            btfsc  STATUS,Z
            goto   Do_Tmr1X
            incf   Tmr1_Pres,F
            movlw  D'25'
            subwf  Tmr1_Pres,W
            btfss  STATUS,C
            goto   Do_Tmr1X
            clrf   Tmr1_Pres
            decf   Tmr1_Sec,F
            movf   Tmr1_Sec,W
            andlw  H'0F'
            sublw  H'F'
            btfss  STATUS,Z
            goto   Do_Tmr1X
            movlw  D'6'
            subwf  Tmr1_Sec,F
Do_Tmr1X:return

;LCD Subroutine

LCD_DPort     equu    PORTB
LCD_CPort     equ    PORTD
LCD_EN        equ    7
LCD_RS        equ    6

LCD_Line_Max  equ    D'2'
LCD_Char_Max equ    D'16'
LCD_L1_Addr   equ    D'00' +H'80'
LCD_L2_Addr   equ    LCD_L1_Addr +D'40'

Set_RS0:macro
            bcf    LCD_CPort,LCD_RS
            endm

Set_RS1: macro
            bsf    LCD_CPort,LCD_RS
            endm

Pulse_EN:bsf    LCD_CPort,LCD_EN
            nop
            nop
            nop
            nop
            bcf    LCD_CPort,LCD_EN
            call   Wait1
            return

Init_LCD: Set_RS0
            movlw  D'200'
            call   Wait2
            movlw  D'200'
            call   Wait2

            movlw  H'38'
            movwf  LCD_DPort
            call   Pulse_EN
            movlw  D'100'
            call   Wait2

            call   Pulse_EN
            movlw  D'100'
            call   Wait2

            call   Pulse_EN
            movlw  D'100'
            call   Wait2

            movlw  H'06'
            movwf  LCD_DPort
            call   Pulse_EN

            movlw  H'0F'
            movlw  H'0C'
            movwf  LCD_DPort
            call   Pulse_EN

            movlw  H'14'
            movwf  LCD_DPort
            call   Pulse_EN

            movlw  H'01'
            movwf  LCD_DPort
            call   Pulse_EN

            movlw  D'100'
            call   Wait2

            return
```

```
Wait1:     movlw  H'10'
           movwf  Wait1_Val
Wait1_loop:decf   Wait1_Val,F
           btfss  STATUS,Z
           goto   Wait1_loop
           return

Wait2:     movwf  Wait2_Val
Wait2_loop    call   Wait1
           decf   Wait2_Val,F
           btfss  STATUS,Z
           goto   Wait2_loop
           return

Disp_LCD:

Disp_LCD1:Set_RS0
           movlw  LCD_L1_Addr
           movwf  LCD_DPort
           call   Pulse_EN
           Set_RS1
           clrf   Temp1
RAM2LCD1:movlw  LCD_Char_Max
           subwf  Temp1,W
           btfsc  STATUS,Z
           goto   RAM2LCD1X
           movlw  LCD_RAM_Buf
           addwf  Temp1,W
           movwf  FSR
           bsf    FSR,7
           movf   INDF,W
           movwf  LCD_DPort
           call   Pulse_EN
           incf   Temp1,F
           goto   RAM2LCD1
RAM2LCD1X:nop

Disp_LCD2:Set_RS0
           movlw  LCD_L2_Addr
           movwf  LCD_DPort
           call   Pulse_EN
           Set_RS1
           clrf   Temp1
RAM2LCD2:movlw  LCD_Char_Max
           subwf  Temp1,W
           btfsc  STATUS,Z
           goto   RAM2LCD2X
           movlw  LCD_RAM_Buf
           addlw  LCD_Char_Max
           addwf  Temp1,W
           movwf  FSR
           bsf    FSR,7
           movf   INDF,W
           movwf  LCD_DPort
           call   Pulse_EN
           incf   Temp1,F
           goto   RAM2LCD2
RAM2LCD2X:nop
           return

Ld_Msg2RAM:clrf   Temp1
           clrf   Temp3
           movf   Msg_Num,W
           movwf  Temp1

Ld_Msg_Adr:movf   Temp1,W
           btfsc  STATUS,Z
           goto   Ld_MsgLoop
           movlw  D'32'
           addwf  Temp3,F
           decf   Temp1,F
           goto   Ld_Msg_Adr

Ld_MsgLoop:movlw  D'32'
           subwf  Temp1,W
           btfsc  STATUS,Z
           goto   Ld_MsgDone

           movf   PCLATH,W
           movwf  Temp4
           movlw  HIGH Msg0
           movwf  PCLATH
           movf   Temp1,W
           addwf  Temp3,W
           call   Msg0
           movwf  Temp2
           movf   Temp4,W
           movwf  PCLATH
           goto   Ld_Msg_Char

Ld_Msg_Char:movlw  LCD_RAM_Buf
           addwf  Temp1,W
           movwf  FSR
           bsf    FSR,7
           movf   Temp2,W
           movwf  INDF
           incf   Temp1,F
           goto   Ld_MsgLoop
Ld_MsgDone:return

           include <Chord.inc>

           end

Disp_Chord:movf   Tmr1_Sec,W
```

```
        btfss  STATUS,Z
        goto   Disp_ChordX

        clrf   Temp1
        movlw  LCD_RAM_Buf
        addlw  D'16'
        movwf  FSR
        bsf    FSR,7

        call   Disp_C7          ;7th (4/8)
        call   Disp_Db7
        call   Disp_D7
        call   Disp_Eb7
        call   Disp_E7
        call   Disp_F7
        call   Disp_Gb7
        call   Disp_G7
        call   Disp_Ab7
        call   Disp_A7
        call   Disp_Bb7
        call   Disp_B7

        call   Disp_CM7         ;M7th (5/8)
        call   Disp_DbM7
        call   Disp_DM7
        call   Disp_EbM7
        call   Disp_EM7
        call   Disp_FM7
        call   Disp_GbM7
        call   Disp_GM7
        call   Disp_AbM7
        call   Disp_AM7
        call   Disp_BbM7
        call   Disp_BM7

        movf   PCLATH,W
        movwf  PCLATH_TEMP
        bcf    PCLATH,4
        bsf    PCLATH,3

        call   Disp_Cm7         ;m7th              (3/8)
(6/8)
        call   Disp_Dbm7
        call   Disp_Dm7
        call   Disp_Ebm7
        call   Disp_Em7
        call   Disp_Fm7
        call   Disp_Gbm7
        call   Disp_Gm7
        call   Disp_Abm7
        call   Disp_Am7
        call   Disp_Bbm7
        call   Disp_Bm7
```

```
        call   Disp_Cdim        ;dim (7/8)
        call   Disp_Dbdim
        call   Disp_Ddim

        call   Disp_Caug        ;aug (8/8)
        call   Disp_Dbaug
        call   Disp_Daug
        call   Disp_Ebaug

        movf   PCLATH_TEMP,W
        movwf  PCLATH

        call   Disp_C           ;Major (1/8)
        call   Disp_Db
        call   Disp_D
        call   Disp_Eb
        call   Disp_E
        call   Disp_F
        call   Disp_Gb
        call   Disp_G
        call   Disp_Ab
        call   Disp_A
        call   Disp_Bb
        call   Disp_B

        call   Disp_Cm          ;minor (2/8)
        call   Disp_Dbm
        call   Disp_Dm
        call   Disp_Ebm
        call   Disp_Em
        call   Disp_Fm
        call   Disp_Gbm
        call   Disp_Gm
        call   Disp_Abm
        call   Disp_Am
        call   Disp_Bbm
        call   Disp_Bm

        call   Disp_Cs          ;suspended

        call   Disp_Dbs
        call   Disp_Ds
        call   Disp_Ebs
        call   Disp_Es
        call   Disp_Fs
        call   Disp_Gbs
        call   Disp_Gs
        call   Disp_Abs
        call   Disp_As
        call   Disp_Bbs
        call   Disp_Bs
```

```
            call   Disp_None

Disp_ChordX:  return

;'Major Note'

Disp_C:     btfsc  Temp1,0
            goto   Disp_CX
            btfss  Note_Lo,3
            goto   Disp_CX
            btfss  Note_Hi,1
            goto   Disp_CX
            btfss  Note_Hi,4
            goto   Disp_CX
            bsf    Temp1,0
            movlw  "C"
            movwf  INDF
            incf   FSR,F
            call   Disp_Major
Disp_CX: return

Disp_Db: btfsc  Temp1,0
            goto   Disp_DbX
            btfss  Note_Lo,4
            goto   Disp_DbX
            btfss  Note_Hi,2
            goto   Disp_DbX
            btfss  Note_Hi,5
            goto   Disp_DbX
            bsf    Temp1,0
            movlw  "C"
            movwf  INDF
            incf   FSR,F
            movlw  "#"
            movwf  INDF
            incf   FSR,F
            movlw  "/"
            movwf  INDF
            incf   FSR,F
            movlw  "D"
            movwf  INDF
            incf   FSR,F
            movlw  "b"
            movwf  INDF
            incf   FSR,F
            call   Disp_Major
Disp_DbX:return

Disp_D:     btfsc  Temp1,0
            goto   Disp_DX
            btfss  Note_Lo,5
            goto   Disp_DX
            btfss  Note_Hi,3
            goto   Disp_DX
            btfss  Note_Lo,0
            goto   Disp_DX
            bsf    Temp1,0
            movlw  "D"
            movwf  INDF
            incf   FSR,F
            call   Disp_Major
Disp_DX: return

Disp_Eb:  btfsc  Temp1,0
            goto   Disp_EbX
            btfss  Note_Hi,0
            goto   Disp_EbX
            btfss  Note_Hi,4
            goto   Disp_EbX
            btfss  Note_Lo,1
            goto   Disp_EbX
            bsf    Temp1,0
            movlw  "D"
            movwf  INDF
            incf   FSR,F
            movlw  "#"
            movwf  INDF
            incf   FSR,F
            movlw  "/"
            movwf  INDF
            incf   FSR,F
            movlw  "E"
            movwf  INDF
            incf   FSR,F
            movlw  "b"
            movwf  INDF
            incf   FSR,F
            call   Disp_Major
Disp_EbX:return

Disp_E:     btfsc  Temp1,0
            goto   Disp_EX
            btfss  Note_Hi,1
            goto   Disp_EX
            btfss  Note_Hi,5
            goto   Disp_EX
            btfss  Note_Lo,2
            goto   Disp_EX
            bsf    Temp1,0
            movlw  "E"
            movwf  INDF
            incf   FSR,F
            call   Disp_Major
Disp_EX: return

Disp_F:     btfsc  Temp1,0
```

```
          goto   Disp_FX
          btfss  Note_Hi,2
          goto   Disp_FX
          btfss  Note_Lo,0
          goto   Disp_FX
          btfss  Note_Lo,3
          goto   Disp_FX
          bsf    Temp1,0
          movlw  "F"
          movwf  INDF
          incf   FSR,F
          call   Disp_Major
Disp_FX:  return

Disp_Gb: btfsc  Temp1,0
          goto   Disp_GbX
          btfss  Note_Hi,3
          goto   Disp_GbX
          btfss  Note_Lo,1
          goto   Disp_GbX
          btfss  Note_Lo,4
          goto   Disp_GbX
          bsf    Temp1,0
          movlw  "F"
          movwf  INDF
          incf   FSR,F
          movlw  "#"
          movwf  INDF
          incf   FSR,F
          movlw  "/"
          movwf  INDF
          incf   FSR,F
          movlw  "G"
          movwf  INDF
          incf   FSR,F
          movlw  "b"
          movwf  INDF
          incf   FSR,F
          call   Disp_Major
Disp_GbX:return

Disp_G:   btfsc  Temp1,0
          goto   Disp_GX
          btfss  Note_Hi,4
          goto   Disp_GX
          btfss  Note_Lo,2
          goto   Disp_GX
          btfss  Note_Lo,5
          goto   Disp_GX
          bsf    Temp1,0
          movlw  "G"
          movwf  INDF
          incf   FSR,F
```

```
          call   Disp_Major
Disp_GX: return
Disp_Ab: btfsc  Temp1,0
          goto   Disp_AbX
          btfss  Note_Hi,5
          goto   Disp_AbX
          btfss  Note_Lo,3
          goto   Disp_AbX
          btfss  Note_Hi,0
          goto   Disp_AbX
          bsf    Temp1,0
          movlw  "G"
          movwf  INDF
          incf   FSR,F
          movlw  "#"
          movwf  INDF
          incf   FSR,F
          movlw  "/"
          movwf  INDF
          incf   FSR,F
          movlw  "A"
          movwf  INDF
          incf   FSR,F
          movlw  "b"
          movwf  INDF
          incf   FSR,F
          call   Disp_Major
Disp_AbX: return

Disp_A:   btfsc  Temp1,0
          goto   Disp_AX
          btfss  Note_Lo,0
          goto   Disp_AX
          btfss  Note_Lo,4
          goto   Disp_AX
          btfss  Note_Hi,1
          goto   Disp_AX
          bsf    Temp1,0
          movlw  "A"
          movwf  INDF
          incf   FSR,F
          call   Disp_Major
Disp_AX: return

Disp_Bb: btfsc  Temp1,0
          goto   Disp_BbX
          btfss  Note_Lo,1
          goto   Disp_BbX
          btfss  Note_Lo,5
          goto   Disp_BbX
          btfss  Note_Hi,2
          goto   Disp_BbX
          bsf    Temp1,0
          movlw  "A"
          movwf  INDF
          incf   FSR,F
          movlw  "#"
          movwf  INDF
          incf   FSR,F
          movlw  "/"
          movwf  INDF
          incf   FSR,F
          movlw  "B"
          movwf  INDF
          incf   FSR,F
          movlw  "b"
          movwf  INDF
          incf   FSR,F
          call   Disp_Major
Disp_BbX:return

Disp_B:   btfsc  Temp1,0
          goto   Disp_BX
          btfss  Note_Lo,2
          goto   Disp_BX
          btfss  Note_Hi,0
          goto   Disp_BX
          btfss  Note_Hi,3
          goto   Disp_BX
          bsf    Temp1,0
          movlw  "B"
          movwf  INDF
          incf   FSR,F
          call   Disp_Major
Disp_BX:      return

;'minor note'
Disp_Cm: btfsc  Temp1,0
          goto   Disp_CmX
          btfss  Note_Lo,3
          goto   Disp_CmX
          btfss  Note_Hi,0
          goto   Disp_CmX
          btfss  Note_Hi,4
          goto   Disp_CmX
          bsf    Temp1,0
          movlw  "C"
          movwf  INDF
          incf   FSR,F
          movlw  "m"
          movwf  INDF
          incf   FSR,F
          call   Disp_Minor
Disp_CmX: return

Disp_Dbm:btfsc  Temp1,0
          goto   Disp_DbmX
          btfss  Note_Lo,4
          goto   Disp_DbmX
          btfss  Note_Hi,1
          goto   Disp_DbmX
          btfss  Note_Hi,5
          goto   Disp_DbmX
          bsf    Temp1,0
          movlw  "C"
          movwf  INDF
          incf   FSR,F
          movlw  "#"
          movwf  INDF
          incf   FSR,F
          movlw  "m"
          movwf  INDF
          incf   FSR,F
          movlw  "/"
          movwf  INDF
          incf   FSR,F
          movlw  "D"
          movwf  INDF
          incf   FSR,F
          movlw  "b"
          movwf  INDF
          incf   FSR,F
          movlw  "m"
          movwf  INDF
          incf   FSR,F
          call   Disp_Minor
Disp_DbmX:return

Disp_Dm:btfsc  Temp1,0
          goto   Disp_DmX
          btfss  Note_Lo,5
          goto   Disp_DmX
          btfss  Note_Hi,2
          goto   Disp_DmX
          btfss  Note_Lo,0
          goto   Disp_DmX
          bsf    Temp1,0
          movlw  "D"
          movwf  INDF
          incf   FSR,F
          movlw  "m"
          movwf  INDF
          incf   FSR,F
          call   Disp_Minor
Disp_DmX: return

Disp_Ebm: btfsc  Temp1,0
          goto   Disp_EbmX
          btfss  Note_Hi,0
```

```
            goto   Disp_EbmX
            btfss  Note_Hi,3
            goto   Disp_EbmX
            btfss  Note_Lo,1
            goto   Disp_EbmX
            bsf    Temp1,0
            movlw  "D"
            movwf  INDF
            incf   FSR,F
            movlw  "#"
            movwf  INDF
            incf   FSR,F
            movlw  "m"
            movwf  INDF
            incf   FSR,F
            movlw  "/"
            movwf  INDF
            incf   FSR,F
            movlw  "E"
            movwf  INDF
            incf   FSR,F
            movlw  "b"
            movwf  INDF
            incf   FSR,F
            movlw  "m"
            movwf  INDF
            incf   FSR,F
            call   Disp_Minor
Disp_EbmX:return

Disp_Em: btfsc  Temp1,0
            goto   Disp_EmX
            btfss  Note_Hi,1
            goto   Disp_EmX
            btfss  Note_Hi,4
            goto   Disp_EmX
            btfss  Note_Lo,2
            goto   Disp_EmX
            bsf    Temp1,0
            movlw  "E"
            movwf  INDF
            incf   FSR,F
            movlw  "m"
            movwf  INDF
            incf   FSR,F
            call   Disp_Minor
Disp_EmX:return

Disp_Fm:btfsc  Temp1,0
            goto   Disp_FmX
            btfss  Note_Hi,2
            goto   Disp_FmX
            btfss  Note_Hi,5
```

```
            goto   Disp_FmX
            btfss  Note_Lo,3
            goto   Disp_FmX
            bsf    Temp1,0
            movlw  "F"
            movwf  INDF
            incf   FSR,F
            movlw  "m"
            movwf  INDF
            incf   FSR,F
            call   Disp_Minor
Disp_FmX:return

Disp_Gbm:btfsc  Temp1,0
            goto   Disp_GbmX
            btfss  Note_Hi,3
            goto   Disp_GbmX
            btfss  Note_Lo,0
            goto   Disp_GbmX
            btfss  Note_Lo,4
            goto   Disp_GbmX
            bsf    Temp1,0
            movlw  "F"
            movwf  INDF
            incf   FSR,F
            movlw  "#"
            movwf  INDF
            incf   FSR,F
            movlw  "m"
            movwf  INDF
            incf   FSR,F
            movlw  "/"
            movwf  INDF
            incf   FSR,F
            movlw  "G"
            movwf  INDF
            incf   FSR,F
            movlw  "b"
            movwf  INDF
            incf   FSR,F
            movlw  "m"
            movwf  INDF
            incf   FSR,F
            call   Disp_Minor
Disp_GbmX:return

Disp_Gm:btfsc  Temp1,0
            goto   Disp_GmX
            btfss  Note_Hi,4
            goto   Disp_GmX
            btfss  Note_Lo,1
            goto   Disp_GmX
            btfss  Note_Lo,5
            goto   Disp_GmX
            bsf    Temp1,0
            movlw  "G"
            movwf  INDF
            incf   FSR,F
            movlw  "m"
            movwf  INDF
            incf   FSR,F
            call   Disp_Minor
Disp_GmX:return

Disp_Abm:btfsc  Temp1,0
            goto   Disp_AbmX
            btfss  Note_Hi,5
            goto   Disp_AbmX
            btfss  Note_Lo,2
            goto   Disp_AbmX
            btfss  Note_Hi,0
            goto   Disp_AbmX
            bsf    Temp1,0
            movlw  "G"
            movwf  INDF
            incf   FSR,F
            movlw  "#"
            movwf  INDF
            incf   FSR,F
            movlw  "m"
            movwf  INDF
            incf   FSR,F
            movlw  "/"
            movwf  INDF
            incf   FSR,F
            movlw  "A"
            movwf  INDF
            incf   FSR,F
            movlw  "b"
            movwf  INDF
            incf   FSR,F
            movlw  "m"
            movwf  INDF
            incf   FSR,F
            call   Disp_Minor
Disp_AbmX:return

Disp_Am: btfsc  Temp1,0
            goto   Disp_AmX
            btfss  Note_Lo,0
            goto   Disp_AmX
            btfss  Note_Lo,3
            goto   Disp_AmX
            btfss  Note_Hi,1
            goto   Disp_AmX
            bsf    Temp1,0
```

```
            movlw  "A"
            movwf  INDF
            incf   FSR,F
            movlw  "m"
            movwf  INDF
            incf   FSR,F
            call   Disp_Minor
Disp_AmX: return

Disp_Bbm:btfsc  Temp1,0
            goto   Disp_BbmX
            btfss  Note_Lo,1
            goto   Disp_BbmX
            btfss  Note_Lo,4
            goto   Disp_BbmX
            btfss  Note_Hi,2
            goto   Disp_BbmX
            bsf    Temp1,0
            movlw  "A"
            movwf  INDF
            incf   FSR,F
            movlw  "#"
            movwf  INDF
            incf   FSR,F
            movlw  "m"
            movwf  INDF
            incf   FSR,F
            movlw  "/"
            movwf  INDF
            incf   FSR,F
            movlw  "B"
            movwf  INDF
            incf   FSR,F
            movlw  "b"
            movwf  INDF
            incf   FSR,F
            movlw  "m"
            movwf  INDF
            incf   FSR,F
            call   Disp_Minor
Disp_BbmX:return

Disp_Bm:btfsc  Temp1,0
            goto   Disp_BmX
            btfss  Note_Lo,2
            goto   Disp_BmX
            btfss  Note_Lo,5
            goto   Disp_BmX
            btfss  Note_Hi,3
            goto   Disp_BmX
            bsf    Temp1,0
            movlw  "B"
            movwf  INDF
```

```
          incf   FSR,F
          movlw  "m"
          movwf  INDF
          incf   FSR,F
          call   Disp_Minor
Disp_BmX:return

Disp_Major:movlw  " "
          movwf  INDF
          incf   FSR,F
          movlw  "M"
          movwf  INDF
          incf   FSR,F
          movlw  "a"
          movwf  INDF
          incf   FSR,F
          movlw  "j"
          movwf  INDF
          incf   FSR,F
          movlw  "o"
          movwf  INDF
          incf   FSR,F
          movlw  "r"
          movwf  INDF
          incf   FSR,F
Disp_MajorX:return

Disp_Minor:movlw  " "
          movwf  INDF
          incf   FSR,F
          movlw  "M"
          movwf  INDF
          incf   FSR,F
          movlw  "i"
          movwf  INDF
          incf   FSR,F
          movlw  "n"
          movwf  INDF
          incf   FSR,F
          movlw  "o"
          movwf  INDF
          incf   FSR,F
          movlw  "r"
          movwf  INDF
          incf   FSR,F
Disp_MinorX:return

Disp_None:btfsc  Temp1,0
          goto   Disp_NoneX
          movlw  "T"
          movwf  INDF
          incf   FSR,F
          movlw  "r"
          movwf  INDF
          incf   FSR,F
          movlw  "y"
          movwf  INDF
          incf   FSR,F
          movlw  " "
          movwf  INDF
          incf   FSR,F
          movlw  "A"
          movwf  INDF
          incf   FSR,F
          movlw  "g"
          movwf  INDF
          incf   FSR,F
          movlw  "a"
          movwf  INDF
          incf   FSR,F
          movlw  "i"
          movwf  INDF
          incf   FSR,F
          movlw  "n"
          movwf  INDF
          incf   FSR,F
Disp_NoneX:return

          include <Chord2.inc>
          include <Chord3.inc>
;         'suspended Note'
Disp_Cs:  btfsc  Temp1,0
          goto   Disp_CsX
          btfss  Note_Lo,3
          goto   Disp_CsX
          btfss  Note_Hi,2
          goto   Disp_CsX
          btfss  Note_Hi,4
          goto   Disp_CsX
          bsf    Temp1,0
          movlw  "C"
          movwf  INDF
          incf   FSR,F
          call   Disp_sus
Disp_CsX:return

Disp_Dbs:btfsc  Temp1,0
          goto   Disp_DbsX
          btfss  Note_Lo,4
          goto   Disp_DbsX
          btfss  Note_Hi,3
          goto   Disp_DbsX
          btfss  Note_Hi,5
          goto   Disp_DbsX
          bsf    Temp1,0
          movlw  "C"
          movwf  INDF
          incf   FSR,F
          movlw  "#"
          movwf  INDF
          incf   FSR,F
          call   Disp_sus
          movlw  "/"
          movwf  INDF
          incf   FSR,F
          movlw  "D"
          movwf  INDF
          incf   FSR,F
          movlw  "b"
          movwf  INDF
          incf   FSR,F
          call   Disp_sus
Disp_DbsX:return

Disp_Ds: btfsc  Temp1,0
          goto   Disp_DsX
          btfss  Note_Lo,5
          goto   Disp_DsX
          btfss  Note_Hi,4
          goto   Disp_DsX
          btfss  Note_Lo,0
          goto   Disp_DsX
          bsf    Temp1,0
          movlw  "D"
          movwf  INDF
          incf   FSR,F
          call   Disp_sus
Disp_DsX:return

Disp_Ebs:btfsc  Temp1,0
          goto   Disp_EbsX
          btfss  Note_Hi,0
          goto   Disp_EbsX
          btfss  Note_Hi,5
          goto   Disp_EbsX
          btfss  Note_Lo,1
          goto   Disp_EbsX
          bsf    Temp1,0
          movlw  "D"
          movwf  INDF
          incf   FSR,F
          movlw  "#"
          movwf  INDF
          incf   FSR,F
          call   Disp_sus
          movlw  "/"
          movwf  INDF
          incf   FSR,F
          movlw  "E"
```

```
          movwf  INDF
          incf   FSR,F
          movlw  "b"
          movwf  INDF
          incf   FSR,F
          call   Disp_sus
Disp_EbsX: return

Disp_Es:  btfsc  Temp1,0
          goto   Disp_EsX
          btfss  Note_Hi,1
          goto   Disp_EsX
          btfss  Note_Lo,0
          goto   Disp_EsX
          btfss  Note_Lo,2
          goto   Disp_EsX
          bsf    Temp1,0
          movlw  "E"
          movwf  INDF
          incf   FSR,F
          call   Disp_sus
Disp_EsX:return

Disp_Fs:  btfsc  Temp1,0
          goto   Disp_FsX
          btfss  Note_Hi,2
          goto   Disp_FsX
          btfss  Note_Lo,1
          goto   Disp_FsX
          btfss  Note_Lo,3
          goto   Disp_FsX
          bsf    Temp1,0
          movlw  "F"
          movwf  INDF
          incf   FSR,F
          call   Disp_sus
Disp_FsX:return

Disp_Gbs:btfsc  Temp1,0
          goto   Disp_GbsX
          btfss  Note_Hi,3
          goto   Disp_GbsX
          btfss  Note_Lo,2
          goto   Disp_GbsX
          btfss  Note_Lo,4
          goto   Disp_GbsX
          bsf    Temp1,0
          movlw  "F"
          movwf  INDF
          incf   FSR,F
          movlw  "#"
          movwf  INDF
          incf   FSR,F
          call   Disp_sus
          movlw  "/"
          movwf  INDF
          incf   FSR,F
          movlw  "G"
          movwf  INDF
          incf   FSR,F
          movlw  "b"
          movwf  INDF
          incf   FSR,F
          call   Disp_sus
Disp_GbsX:return

Disp_Gs: btfsc  Temp1,0
          goto   Disp_GsX
          btfss  Note_Hi,4
          goto   Disp_GsX
          btfss  Note_Lo,3
          goto   Disp_GsX
          btfss  Note_Lo,5
          goto   Disp_GsX
          bsf    Temp1,0
          movlw  "G"
          movwf  INDF
          incf   FSR,F
          call   Disp_sus
Disp_GsX:return

Disp_Abs:btfsc  Temp1,0
          goto   Disp_AbsX
          btfss  Note_Hi,5
          goto   Disp_AbsX
          btfss  Note_Lo,4
          goto   Disp_AbsX
          btfss  Note_Hi,0
          goto   Disp_AbsX
          bsf    Temp1,0
          movlw  "G"
          movwf  INDF
          incf   FSR,F
          movlw  "#"
          movwf  INDF
          incf   FSR,F
          call   Disp_sus
          movlw  "/"
          movwf  INDF
          incf   FSR,F
          movlw  "A"
          movwf  INDF
          incf   FSR,F
          movlw  "b"
          movwf  INDF
          incf   FSR,F
```

```
          call   Disp_sus
Disp_AbsX:return

Disp_As: btfsc  Temp1,0
          goto   Disp_AsX
          btfss  Note_Lo,0
          goto   Disp_AsX
          btfss  Note_Lo,5
          goto   Disp_AsX
          btfss  Note_Hi,1
          goto   Disp_AsX
          bsf    Temp1,0
          movlw  "A"
          movwf  INDF
          incf   FSR,F
          call   Disp_sus
Disp_AsX:return

Disp_Bbs: btfsc  Temp1,0
          goto   Disp_BbsX
          btfss  Note_Lo,1
          goto   Disp_BbsX
          btfss  Note_Hi,0
          goto   Disp_BbsX
          btfss  Note_Hi,2
          goto   Disp_BbsX
          bsf    Temp1,0
          movlw  "A"
          movwf  INDF
          incf   FSR,F
          movlw  "#"
          movwf  INDF
          incf   FSR,F
          call   Disp_sus
          movlw  "/"
          movwf  INDF
          incf   FSR,F
          movlw  "B"
          movwf  INDF
          incf   FSR,F
          movlw  "b"
          movwf  INDF
          incf   FSR,F
          call   Disp_sus
Disp_BbsX:return

Disp_Bs: btfsc  Temp1,0
          goto   Disp_BsX
          btfss  Note_Lo,2
          goto   Disp_BsX
          btfss  Note_Hi,1
          goto   Disp_BsX
          btfss  Note_Hi,3
          goto   Disp_BsX
          bsf    Temp1,0
          movlw  "B"
          movwf  INDF
          incf   FSR,F
          call   Disp_sus
Disp_BsX:return

Disp_sus:movlw  "s"
          movwf  INDF
          incf   FSR,F
          movlw  "u"
          movwf  INDF
          incf   FSR,F
          movlw  "s"
          movwf  INDF
          incf   FSR,F
          return

;'7th Note'
Disp_C7: btfsc  Temp1,0
          goto   Disp_C7X
          btfss  Note_Lo,3
          goto   Disp_C7X
          btfss  Note_Hi,1
          goto   Disp_C7X
          btfss  Note_Hi,4
          goto   Disp_C7X
          btfss  Note_Lo,1
          goto   Disp_C7X
          bsf    Temp1,0
          movlw  "C"
          movwf  INDF
          incf   FSR,F
          call   Disp_7
Disp_C7X:return

Disp_Db7:btfsc  Temp1,0
          goto   Disp_Db7X
          btfss  Note_Lo,4
          goto   Disp_Db7X
          btfss  Note_Hi,2
          goto   Disp_Db7X
          btfss  Note_Hi,5
          goto   Disp_Db7X
          btfss  Note_Lo,2
          goto   Disp_Db7X
          bsf    Temp1,0
          movlw  "C"
          movwf  INDF
          incf   FSR,F
          movlw  "#"
          movwf  INDF
          incf   FSR,F
          call   Disp_7
          movlw  "/"
          movwf  INDF
          incf   FSR,F
          movlw  "D"
          movwf  INDF
          incf   FSR,F
          movlw  "b"
          movwf  INDF
          incf   FSR,F
          call   Disp_7
Disp_Db7X:return

Disp_D7: btfsc  Temp1,0
          goto   Disp_D7X
          btfss  Note_Lo,5
          goto   Disp_D7X
          btfss  Note_Hi,3
          goto   Disp_D7X
          btfss  Note_Lo,0
          goto   Disp_D7X
          btfss  Note_Lo,3
          goto   Disp_D7X
          bsf    Temp1,0
          movlw  "D"
          movwf  INDF
          incf   FSR,F
          call   Disp_7
Disp_D7X:return

Disp_Eb7:btfsc  Temp1,0
          goto   Disp_Eb7X
          btfss  Note_Hi,0
          goto   Disp_Eb7X
          btfss  Note_Hi,4
          goto   Disp_Eb7X
          btfss  Note_Lo,1
          goto   Disp_Eb7X
          btfss  Note_Lo,4
          goto   Disp_Eb7X
          bsf    Temp1,0
          movlw  "D"
          movwf  INDF
          incf   FSR,F
          movlw  "#"
          movwf  INDF
          incf   FSR,F
          call   Disp_7
          movlw  "/"
          movwf  INDF
          incf   FSR,F
          movlw  "E"
```

```
        movwf  INDF
        incf   FSR,F
        movlw  "b"
        movwf  INDF
        incf   FSR,F
        call   Disp_7
Disp_Eb7X:return

Disp_E7: btfsc  Temp1,0
        goto   Disp_E7X
        btfss  Note_Hi,1
        goto   Disp_E7X
        btfss  Note_Hi,5
        goto   Disp_E7X
        btfss  Note_Lo,2
        goto   Disp_E7X
        btfss  Note_Lo,5
        goto   Disp_E7X
        bsf    Temp1,0
        movlw  "E"
        movwf  INDF
        incf   FSR,F
        call   Disp_7
Disp_E7X:return

Disp_F7: btfsc  Temp1,0
        goto   Disp_F7X
        btfss  Note_Hi,2
        goto   Disp_F7X
        btfss  Note_Lo,0
        goto   Disp_F7X
        btfss  Note_Lo,3
        goto   Disp_F7X
        btfss  Note_Hi,0
        goto   Disp_F7X
        bsf    Temp1,0
        movlw  "F"
        movwf  INDF
        incf   FSR,F
        call   Disp_7
Disp_F7X:return

Disp_Gb7:btfsc  Temp1,0
        goto   Disp_Gb7X
        btfss  Note_Hi,3
        goto   Disp_Gb7X
        btfss  Note_Lo,1
        goto   Disp_Gb7X
        btfss  Note_Lo,4
        goto   Disp_Gb7X
        btfss  Note_Hi,1
        goto   Disp_Gb7X
        bsf    Temp1,0
```

```
        movlw  "F"
        movwf  INDF
        incf   FSR,F
        movlw  "#"
        movwf  INDF
        incf   FSR,F
        call   Disp_7
        movlw  "/"
        movwf  INDF
        incf   FSR,F
        movlw  "G"
        movwf  INDF
        incf   FSR,F
        movlw  "b"
        movwf  INDF
        incf   FSR,F
        call   Disp_7
Disp_Gb7X:return

Disp_G7: btfsc  Temp1,0
        goto   Disp_G7X
        btfss  Note_Hi,4
        goto   Disp_G7X
        btfss  Note_Lo,2
        goto   Disp_G7X
        btfss  Note_Lo,5
        goto   Disp_G7X
        btfss  Note_Hi,2
        goto   Disp_G7X
        bsf    Temp1,0
        movlw  "G"
        movwf  INDF
        incf   FSR,F
        call   Disp_7
Disp_G7X:return

Disp_Ab7:btfsc  Temp1,0
        goto   Disp_Ab7X
        btfss  Note_Hi,5
        goto   Disp_Ab7X
        btfss  Note_Lo,3
        goto   Disp_Ab7X
        btfss  Note_Hi,0
        goto   Disp_Ab7X
        btfss  Note_Hi,3
        goto   Disp_Ab7X
        bsf    Temp1,0
        movlw  "G"
        movwf  INDF
        incf   FSR,F
        movlw  "#"
        movwf  INDF
        incf   FSR,F
```

```
            call   Disp_7
            movlw  "/"
            movwf  INDF
            incf   FSR,F
            movlw  "A"
            movwf  INDF
            incf   FSR,F
            movlw  "b"
            movwf  INDF
            incf   FSR,F
            call   Disp_7
Disp_Ab7X:return

Disp_A7: btfsc  Temp1,0
            goto   Disp_A7X
            btfss  Note_Lo,0
            goto   Disp_A7X
            btfss  Note_Lo,4
            goto   Disp_A7X
            btfss  Note_Hi,1
            goto   Disp_A7X
            btfss  Note_Hi,4
            goto   Disp_A7X
            bsf    Temp1,0
            movlw  "A"
            movwf  INDF
            incf   FSR,F
            call   Disp_7
Disp_A7X: return

Disp_Bb7:btfsc  Temp1,0
            goto   Disp_Bb7X
            btfss  Note_Lo,1
            goto   Disp_Bb7X
            btfss  Note_Lo,5
            goto   Disp_Bb7X
            btfss  Note_Hi,2
            goto   Disp_Bb7X
            btfss  Note_Hi,5
            goto   Disp_Bb7X
            bsf    Temp1,0
            movlw  "A"
            movwf  INDF
            incf   FSR,F
            movlw  "#"
            movwf  INDF
            incf   FSR,F
            call   Disp_7
            movlw  "/"
            movwf  INDF
            incf   FSR,F
            movlw  "B"
            movwf  INDF
            incf   FSR,F
            movlw  "b"
            movwf  INDF
            incf   FSR,F
            call   Disp_7
Disp_Bb7X:return

Disp_B7: btfsc  Temp1,0
            goto   Disp_B7X
            btfss  Note_Lo,2
            goto   Disp_B7X
            btfss  Note_Hi,0
            goto   Disp_B7X
            btfss  Note_Hi,3
            goto   Disp_B7X
            btfss  Note_Lo,0
            goto   Disp_B7X
            bsf    Temp1,0
            movlw  "B"
            movwf  INDF
            incf   FSR,F
            call   Disp_7
Disp_B7X:return

Disp_7:   movlw  "7"
            movwf  INDF
            incf   FSR,F
            return

;'M7 Note'

Disp_CM7:btfsc  Temp1,0
            goto   Disp_CM7X
            btfss  Note_Lo,3
            goto   Disp_CM7X
            btfss  Note_Hi,1
            goto   Disp_CM7X
            btfss  Note_Hi,4
            goto   Disp_CM7X
            btfss  Note_Lo,2
            goto   Disp_CM7X
            bsf    Temp1,0
            movlw  "C"
            movwf  INDF
            incf   FSR,F
            call   Disp_M7
Disp_CM7X: return

Disp_DbM7:btfsc  Temp1,0
            goto   Disp_DbM7X
            btfss  Note_Lo,4
            goto   Disp_DbM7X
            btfss  Note_Hi,2
```

```
          goto   Disp_DbM7X
          btfss  Note_Hi,5
          goto   Disp_DbM7X
          btfss  Note_Lo,3
          goto   Disp_DbM7X
          bsf    Temp1,0
          movlw  "C"
          movwf  INDF
          incf   FSR,F
          movlw  "#"
          movwf  INDF
          incf   FSR,F
          call   Disp_M7
          movlw  "/"
          movwf  INDF
          incf   FSR,F
          movlw  "D"
          movwf  INDF
          incf   FSR,F
          movlw  "b"
          movwf  INDF
          incf   FSR,F
          call   Disp_M7
Disp_DbM7X:return

Disp_DM7:btfsc  Temp1,0
          goto   Disp_DM7X
          btfss  Note_Lo,5
          goto   Disp_DM7X
          btfss  Note_Hi,3
          goto   Disp_DM7X
          btfss  Note_Lo,0
          goto   Disp_DM7X
          btfss  Note_Lo,4
          goto   Disp_DM7X
          bsf    Temp1,0
          movlw  "D"
          movwf  INDF
          incf   FSR,F
          call   Disp_M7
Disp_DM7X:return

Disp_EbM7:btfsc  Temp1,0
          goto   Disp_EbM7X
          btfss  Note_Hi,0
          goto   Disp_EbM7X
          btfss  Note_Hi,4
          goto   Disp_EbM7X
          btfss  Note_Lo,1
          goto   Disp_EbM7X
          btfss  Note_Lo,5
          goto   Disp_EbM7X
          bsf    Temp1,0
          movlw  "D"
          movwf  INDF
          incf   FSR,F
          movlw  "#"
          movwf  INDF
          incf   FSR,F
          call   Disp_M7
          movlw  "/"
          movwf  INDF
          incf   FSR,F
          movlw  "E"
          movwf  INDF
          incf   FSR,F
          movlw  "b"
          movwf  INDF
          incf   FSR,F
          call   Disp_M7
Disp_EbM7X:return

Disp_EM7:btfsc  Temp1,0
          goto   Disp_EM7X
          btfss  Note_Hi,1
          goto   Disp_EM7X
          btfss  Note_Hi,5
          goto   Disp_EM7X
          btfss  Note_Lo,2
          goto   Disp_EM7X
          btfss  Note_Hi,0
          goto   Disp_EM7X
          bsf    Temp1,0
          movlw  "E"
          movwf  INDF
          incf   FSR,F
          call   Disp_M7
Disp_EM7X:return

Disp_FM7:btfsc  Temp1,0
          goto   Disp_FM7X
          btfss  Note_Hi,2
          goto   Disp_FM7X
          btfss  Note_Lo,0
          goto   Disp_FM7X
          btfss  Note_Lo,3
          goto   Disp_FM7X
          btfss  Note_Hi,1
          goto   Disp_FM7X
          bsf    Temp1,0
          movlw  "F"
          movwf  INDF
          incf   FSR,F
          call   Disp_M7
Disp_FM7X:return
```

```
Disp_GbM7:btfsc  Temp1,0
          goto   Disp_GbM7X
          btfss  Note_Hi,3
          goto   Disp_GbM7X
          btfss  Note_Lo,1
          goto   Disp_GbM7X
          btfss  Note_Lo,4
          goto   Disp_GbM7X
          btfss  Note_Hi,2
          goto   Disp_GbM7X
          bsf    Temp1,0
          movlw  "F"
          movwf  INDF
          incf   FSR,F
          movlw  "#"
          movwf  INDF
          incf   FSR,F
          call   Disp_M7
          movlw  "/"
          movwf  INDF
          incf   FSR,F
          movlw  "G"
          movwf  INDF
          incf   FSR,F
          movlw  "b"
          movwf  INDF
          incf   FSR,F
          call   Disp_M7
Disp_GbM7X:return

Disp_GM7:btfsc  Temp1,0
          goto   Disp_GM7X
          btfss  Note_Hi,4
          goto   Disp_GM7X
          btfss  Note_Lo,2
          goto   Disp_GM7X
          btfss  Note_Lo,5
          goto   Disp_GM7X
          btfss  Note_Hi,3
          goto   Disp_GM7X
          bsf    Temp1,0
          movlw  "G"
          movwf  INDF
          incf   FSR,F
          call   Disp_M7
Disp_GM7X:return

Disp_AbM7:btfsc  Temp1,0
          goto   Disp_AbM7X
          btfss  Note_Hi,5
          goto   Disp_AbM7X
          btfss  Note_Lo,3
          goto   Disp_AbM7X
          btfss  Note_Hi,0
          goto   Disp_AbM7X
          btfss  Note_Hi,4
          goto   Disp_AbM7X
          bsf    Temp1,0
          movlw  "G"
          movwf  INDF
          incf   FSR,F
          movlw  "#"
          movwf  INDF
          incf   FSR,F
          call   Disp_M7
          movlw  "/"
          movwf  INDF
          incf   FSR,F
          movlw  "A"
          movwf  INDF
          incf   FSR,F
          movlw  "b"
          movwf  INDF
          incf   FSR,F
          call   Disp_M7
Disp_AbM7X:return

Disp_AM7:btfsc  Temp1,0
          goto   Disp_AM7X
          btfss  Note_Lo,0
          goto   Disp_AM7X
          btfss  Note_Lo,4
          goto   Disp_AM7X
          btfss  Note_Hi,1
          goto   Disp_AM7X
          btfss  Note_Hi,5
          goto   Disp_AM7X
          bsf    Temp1,0
          movlw  "A"
          movwf  INDF
          incf   FSR,F
          call   Disp_M7
Disp_AM7X:return

Disp_BbM7:btfsc  Temp1,0
          goto   Disp_BbM7X
          btfss  Note_Lo,1
          goto   Disp_BbM7X
          btfss  Note_Lo,5
          goto   Disp_BbM7X
          btfss  Note_Hi,2
          goto   Disp_BbM7X
          btfss  Note_Lo,0
          goto   Disp_BbM7X
          bsf    Temp1,0
          movlw  "A"
```

```
            movwf  INDF
            incf   FSR,F
            movlw  "#"
            movwf  INDF
            incf   FSR,F
            call   Disp_M7
            movlw  "/"
            movwf  INDF
            incf   FSR,F
            movlw  "B"
            movwf  INDF
            incf   FSR,F
            movlw  "b"
            movwf  INDF
            incf   FSR,F
            call   Disp_M7
Disp_BbM7X:return

Disp_BM7:btfsc  Temp1,0
            goto   Disp_BM7X
            btfss  Note_Lo,2
            goto   Disp_BM7X
            btfss  Note_Hi,0
            goto   Disp_BM7X
            btfss  Note_Hi,3
            goto   Disp_BM7X
            btfss  Note_Lo,1
            goto   Disp_BM7X
            bsf    Temp1,0
            movlw  "B"
            movwf  INDF
            incf   FSR,F
            call   Disp_M7
Disp_BM7X:return

Disp_M7: movlw  "M"
            movwf  INDF
            incf   FSR,F
            movlw  "7"
            movwf  INDF
            incf   FSR,F
            return

            org    0x0800

Disp_Cm7:     btfsc  Temp1,0
            goto   Disp_Cm7X
            btfss  Note_Lo,3
            goto   Disp_Cm7X
            btfss  Note_Hi,0
            goto   Disp_Cm7X
            btfss  Note_Hi,4
            goto   Disp_Cm7X
            btfss  Note_Lo,1
            goto   Disp_Cm7X
            bsf    Temp1,0
            movlw  "C"
            movwf  INDF
            incf   FSR,F
            call   Disp_m7
Disp_Cm7X:return

Disp_Dbm7:btfsc  Temp1,0
            goto   Disp_Dbm7X
            btfss  Note_Lo,4
            goto   Disp_Dbm7X
            btfss  Note_Hi,1
            goto   Disp_Dbm7X
            btfss  Note_Hi,5
            goto   Disp_Dbm7X
            btfss  Note_Lo,2
            goto   Disp_Dbm7X
            bsf    Temp1,0
            movlw  "C"
            movwf  INDF
            incf   FSR,F
            movlw  "#"
            movwf  INDF
            incf   FSR,F
            call   Disp_m7
            movlw  "/"
            movwf  INDF
            incf   FSR,F
            movlw  "D"
            movwf  INDF
            incf   FSR,F
            movlw  "b"
            movwf  INDF
            incf   FSR,F
            call   Disp_m7
Disp_Dbm7X:return

Disp_Dm7:btfsc  Temp1,0
            goto   Disp_Dm7X
            btfss  Note_Lo,5
            goto   Disp_Dm7X
            btfss  Note_Hi,2
            goto   Disp_Dm7X
            btfss  Note_Lo,0
            goto   Disp_Dm7X
            btfss  Note_Lo,3
            goto   Disp_Dm7X
            bsf    Temp1,0
            movlw  "D"
            movwf  INDF
            incf   FSR,F
            call   Disp_m7
Disp_Dm7X:return

Disp_Ebm7:btfsc  Temp1,0
            goto   Disp_Ebm7X
            btfss  Note_Hi,0
            goto   Disp_Ebm7X
            btfss  Note_Hi,3
            goto   Disp_Ebm7X
            btfss  Note_Lo,1
            goto   Disp_Ebm7X
            btfss  Note_Lo,4
            goto   Disp_Ebm7X
            bsf    Temp1,0
            movlw  "D"
            movwf  INDF
            incf   FSR,F
            movlw  "#"
            movwf  INDF
            incf   FSR,F
            call   Disp_m7
            movlw  "/"
            movwf  INDF
            incf   FSR,F
            movlw  "E"
            movwf  INDF
            incf   FSR,F
            movlw  "b"
            movwf  INDF
            incf   FSR,F
            call   Disp_m7
Disp_Ebm7X:return

Disp_Em7:btfsc  Temp1,0
            goto   Disp_Em7X
            btfss  Note_Hi,1
            goto   Disp_Em7X
            btfss  Note_Hi,4
            goto   Disp_Em7X
            btfss  Note_Lo,2
            goto   Disp_Em7X
            btfss  Note_Lo,5
            goto   Disp_Em7X
            bsf    Temp1,0
            movlw  "E"
            movwf  INDF
            incf   FSR,F
            call   Disp_m7
Disp_Em7X:return

Disp_Fm7:btfsc  Temp1,0
            goto   Disp_Fm7X
            btfss  Note_Hi,2
            goto   Disp_Fm7X
            btfss  Note_Hi,5
            goto   Disp_Fm7X
            btfss  Note_Lo,3
            goto   Disp_Fm7X
            btfss  Note_Hi,0
            goto   Disp_Fm7X
            bsf    Temp1,0
            movlw  "F"
            movwf  INDF
            incf   FSR,F
            call   Disp_m7
Disp_Fm7X:return

Disp_Gbm7:btfsc  Temp1,0
            goto   Disp_Gbm7X
            btfss  Note_Hi,3
            goto   Disp_Gbm7X
            btfss  Note_Lo,0
            goto   Disp_Gbm7X
            btfss  Note_Lo,4
            goto   Disp_Gbm7X
            btfss  Note_Hi,1
            goto   Disp_Gbm7X
            bsf    Temp1,0
            movlw  "F"
            movwf  INDF
            incf   FSR,F
            movlw  "#"
            movwf  INDF
            incf   FSR,F
            call   Disp_m7
            movlw  "/"
            movwf  INDF
            incf   FSR,F
            movlw  "G"
            movwf  INDF
            incf   FSR,F
            movlw  "b"
            movwf  INDF
            incf   FSR,F
            call   Disp_m7
Disp_Gbm7X:return

Disp_Gm7:btfsc  Temp1,0
            goto   Disp_Gm7X
            btfss  Note_Hi,4
            goto   Disp_Gm7X
            btfss  Note_Lo,1
            goto   Disp_Gm7X
            btfss  Note_Lo,5
            goto   Disp_Gm7X
            btfss  Note_Hi,2
            goto   Disp_Gm7X
            bsf    Temp1,0
            movlw  "G"
            movwf  INDF
            incf   FSR,F
            call   Disp_m7
Disp_Gm7X:return

Disp_Abm7: btfsc  Temp1,0
            goto   Disp_Abm7X
            btfss  Note_Hi,5
            goto   Disp_Abm7X
            btfss  Note_Lo,2
            goto   Disp_Abm7X
            btfss  Note_Hi,0
            goto   Disp_Abm7X
            btfss  Note_Hi,3
            goto   Disp_Abm7X
            bsf    Temp1,0
            movlw  "G"
            movwf  INDF
            incf   FSR,F
            movlw  "#"
            movwf  INDF
            incf   FSR,F
            call   Disp_m7
            movlw  "/"
            movwf  INDF
            incf   FSR,F
            movlw  "A"
            movwf  INDF
            incf   FSR,F
            movlw  "b"
            movwf  INDF
            incf   FSR,F
            call   Disp_m7
Disp_Abm7X:return

Disp_Am7:btfsc  Temp1,0
            goto   Disp_Am7X
            btfss  Note_Lo,0
            goto   Disp_Am7X
            btfss  Note_Lo,3
            goto   Disp_Am7X
            btfss  Note_Hi,1
            goto   Disp_Am7X
            btfss  Note_Hi,4
            goto   Disp_Am7X
            bsf    Temp1,0
            movlw  "A"
            movwf  INDF
            incf   FSR,F
            call   Disp_m7
Disp_Am7X:return

Disp_Bbm7:btfsc  Temp1,0
            goto   Disp_Bbm7X
            btfss  Note_Lo,1
            goto   Disp_Bbm7X
            btfss  Note_Lo,4
            goto   Disp_Bbm7X
            btfss  Note_Hi,2
            goto   Disp_Bbm7X
            btfss  Note_Hi,5
            goto   Disp_Bbm7X
            bsf    Temp1,0
            movlw  "A"
            movwf  INDF
            incf   FSR,F
            movlw  "#"
            movwf  INDF
            incf   FSR,F
            call   Disp_m7
            movlw  "/"
            movwf  INDF
            incf   FSR,F
            movlw  "B"
            movwf  INDF
            incf   FSR,F
            movlw  "b"
            movwf  INDF
            incf   FSR,F
            call   Disp_m7
Disp_Bbm7X:return

Disp_Bm7:btfsc  Temp1,0
            goto   Disp_Bm7X
            btfss  Note_Lo,2
            goto   Disp_Bm7X
            btfss  Note_Lo,5
            goto   Disp_Bm7X
            btfss  Note_Hi,3
            goto   Disp_Bm7X
            btfss  Note_Lo,0
            goto   Disp_Bm7X
            bsf    Temp1,0
            movlw  "B"
            movwf  INDF
            incf   FSR,F
            call   Disp_m7
Disp_Bm7X:return

Disp_m7: movlw  "m"
            movwf  INDF
            incf   FSR,F
            movlw  "7"
            movwf  INDF
            incf   FSR,F
            return

Disp_Cdim:btfsc  Temp1,0
            goto   Disp_CdimX
            btfss  Note_Lo,3
            goto   Disp_CdimX
            btfss  Note_Hi,0
            goto   Disp_CdimX
            btfss  Note_Hi,3
            goto   Disp_CdimX
            btfss  Note_Lo,0
            goto   Disp_CdimX
            bsf    Temp1,0
            movlw  "C"
            movwf  INDF
            incf   FSR,F
            movlw  "."
            movwf  INDF
            incf   FSR,F
            movlw  "D"
            movwf  INDF
            incf   FSR,F
            movlw  "#"
            movwf  INDF
            incf   FSR,F
            movlw  "."
            movwf  INDF
            incf   FSR,F
            movlw  "F"
            movwf  INDF
            incf   FSR,F
            movlw  "#"
            movwf  INDF
            incf   FSR,F
            movlw  "."
            movwf  INDF
            incf   FSR,F
            movlw  "A"
            movwf  INDF
            incf   FSR,F
            movlw  "."
            movwf  INDF
            incf   FSR,F
            call   Disp_dim
Disp_CdimX:return

Disp_Dbdim:btfsc  Temp1,0
            goto   Disp_DbdimX
            btfss  Note_Lo,4
            goto   Disp_DbdimX
            btfss  Note_Hi,1
            goto   Disp_DbdimX
            btfss  Note_Hi,4
            goto   Disp_DbdimX
            btfss  Note_Lo,1
            goto   Disp_DbdimX
            bsf    Temp1,0
            movlw  "C"
            movwf  INDF
            incf   FSR,F
            movlw  "#"
            movwf  INDF
            incf   FSR,F
            movlw  "."
            movwf  INDF
            incf   FSR,F
            movlw  "E"
            movwf  INDF
            incf   FSR,F
            movlw  "."
            movwf  INDF
            incf   FSR,F
            movlw  "G"
            movwf  INDF
            incf   FSR,F
            movlw  "."
            movwf  INDF
            incf   FSR,F
            movlw  "A"
            movwf  INDF
            incf   FSR,F
            movlw  "#"
            movwf  INDF
            incf   FSR,F
            movlw  "."
            movwf  INDF
            incf   FSR,F
            call   Disp_dim
Disp_DbdimX:return

Disp_Ddim:btfsc  Temp1,0
            goto   Disp_DdimX
            btfss  Note_Lo,5
            goto   Disp_DdimX
            btfss  Note_Hi,2
            goto   Disp_DdimX
            btfss  Note_Hi,5
            goto   Disp_DdimX
            btfss  Note_Lo,2
            goto   Disp_DdimX
            bsf    Temp1,0
            movlw  "D"
            movwf  INDF
            incf   FSR,F
            movlw  "."
            movwf  INDF
            incf   FSR,F
            movlw  "F"
            movwf  INDF
            incf   FSR,F
            movlw  "."
            movwf  INDF
            incf   FSR,F
            movlw  "G"
            movwf  INDF
            incf   FSR,F
            movlw  "#"
            movwf  INDF
            incf   FSR,F
            movlw  "."
            movwf  INDF
            incf   FSR,F
            movlw  "B"
            movwf  INDF
            incf   FSR,F
            movlw  "."
            movwf  INDF
            incf   FSR,F
            call   Disp_dim
Disp_DdimX:return

Disp_dim:movlw  "d"
            movwf  INDF
            incf   FSR,F
            movlw  "i"
            movwf  INDF
            incf   FSR,F
            movlw  "m"
            movwf  INDF
            incf   FSR,F
            return

Disp_Caug:btfsc  Temp1,0
            goto   Disp_CaugX
            btfss  Note_Lo,3
            goto   Disp_CaugX
            btfss  Note_Hi,1
            goto   Disp_CaugX
            btfss  Note_Hi,5
            goto   Disp_CaugX
            bsf    Temp1,0
            movlw  "C"
            movwf  INDF
            incf   FSR,F
            movlw  "."
            movwf  INDF
            incf   FSR,F
            movlw  "E"
            movwf  INDF
            incf   FSR,F
            movlw  "."
            movwf  INDF
            incf   FSR,F
            movlw  "G"
            movwf  INDF
            incf   FSR,F
            movlw  "#"
            movwf  INDF
            incf   FSR,F
            movlw  "."
            movwf  INDF
            incf   FSR,F
            call   Disp_aug
Disp_CaugX:return

Disp_Dbaug:btfsc  Temp1,0
            goto   Disp_DbaugX
            btfss  Note_Lo,4
            goto   Disp_DbaugX
            btfss  Note_Hi,2
            goto   Disp_DbaugX
            btfss  Note_Lo,0
            goto   Disp_DbaugX
            bsf    Temp1,0
            movlw  "C"
            movwf  INDF
            incf   FSR,F
            movlw  "#"
            movwf  INDF
            incf   FSR,F
            movlw  "."
            movwf  INDF
            incf   FSR,F
            movlw  "F"
            movwf  INDF
            incf   FSR,F
            movlw  "."
            movwf  INDF
            incf   FSR,F
            movlw  "A"
            movwf  INDF
            incf   FSR,F
            movlw  "."
            movwf  INDF
            incf   FSR,F
            call   Disp_aug
Disp_DbaugX:return

Disp_Daug:btfsc  Temp1,0
            goto   Disp_DaugX
```

```
          btfss  Note_Lo,5
          goto   Disp_DaugX
          btfss  Note_Hi,3
          goto   Disp_DaugX
          btfss  Note_Lo,1
          goto   Disp_DaugX
          bsf    Temp1,0
          movlw  "D"
          movwf  INDF
          incf   FSR,F
          movlw  "."
          movwf  INDF
          incf   FSR,F
          movlw  "F"
          movwf  INDF
          incf   FSR,F
          movlw  "#"
          movwf  INDF
          incf   FSR,F
          movlw  "."
          movwf  INDF
          incf   FSR,F
          movlw  "A"
          movwf  INDF
          incf   FSR,F
          movlw  "#"
          movwf  INDF
          incf   FSR,F
          movlw  "."
          movwf  INDF
          incf   FSR,F
          call   Disp_aug
Disp_DaugX:return

Disp_Ebaug:btfsc  Temp1,0
          goto   Disp_EbaugX
          btfss  Note_Hi,0
          goto   Disp_EbaugX
          btfss  Note_Hi,4
          goto   Disp_EbaugX
          btfss  Note_Lo,2
          goto   Disp_EbaugX
          bsf    Temp1,0
          movlw  "D"
          movwf  INDF
          incf   FSR,F
          movlw  "#"
          movwf  INDF
          incf   FSR,F
          movlw  "."
          movwf  INDF
          incf   FSR,F
          movlw  "G"
          movwf  INDF
          incf   FSR,F
          movlw  "."
          movwf  INDF
          incf   FSR,F
          movlw  "B"
          movwf  INDF
          incf   FSR,F
          movlw  "."
          movwf  INDF
          incf   FSR,F
          call   Disp_aug
Disp_EbaugX:return

Disp_aug:movlw  "a"
          movwf  INDF
          incf   FSR,F
          movlw  "u"
          movwf  INDF
          incf   FSR,F
          movlw  "g"
          movwf  INDF
          incf   FSR,F
          return
```